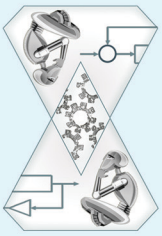


ICINCO 2007

FOURTH INTERNATIONAL CONFERENCE ON INFORMATICS IN CONTROL, AUTOMATION AND ROBOTICS

Proceedings

Intelligent Control Systems and Optimization

ANGERS, FRANCE · MAY 9-12, 2007

ORGANIZED BY

INSTICC

IN COOPERATION WITH

aaai

CO-SPONSORED BY

IFAC

GdR MACS

CNRS

INSTICC PRESS

ICINCO 2007

Proceedings of the
Fourth International Conference on
Informatics in Control, Automation and Robotics

Volume ICSO

Angers, France

May 9 – 12, 2007

Co-organized by
INSTICC – Institute for Systems and Technologies of Information, Control and Communication
and
University of Angers

Co-sponsored by
IFAC - International Federation of Automatic Control
GDR MACS - Groupe de Recherche "Modélisation, Analyse et Conduite des Systèmes dynamiques
CNRS - Centre National de la Recherche Scientifique
and
EEA – Club des Enseignants en Electronique, Electrotechnique et Automatique

In Cooperation with
AAAI – Association for the Advancement of Artificial Intelligence

Edited by Janan Zaytoon, Jean-Louis Ferrier, Juan Andrade Cetto and Joaquim Filipe

Printed in Portugal

ISBN: 978-972-8865-82-5

Depósito Legal: 257879/07

http://www.icinco.org

secretariat@icinco.org

BRIEF CONTENTS

INVITED SPEAKERS

Dimitar Filev

The Ford Motor Company

U.S.A.

Mark W. Spong

University of Illinois at Urbana-Champaign

U.S.A.

Patrick Millot

Université de Valenciennes

France

SPECIAL SESSION CHAIRS

Samir Ladaci

IRCCyN

France

Jean-Louis Boimond

LISA

France

Jean Jacques Loiseau

IRCCyN Nantes

France

Oleg Gusikhin

Ford Research & Adv. Engineering

U.S.A.

ORGANIZING AND STEERING COMMITTEES

Conference Co-chairs

Jean-Louis Ferrier, University of Angers, France

Joaquim Filipe, Polytechnic Institute of Setúbal / INSTICC, Portugal

Program Co-chairs

Juan Andrade Cetto, Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Spain

Janan Zaytoon, CReSTIC, URCA, France

Proceedings Production

Andreia Costa, INSTICC, Portugal

Bruno Encarnação, INSTICC, Portugal

Vitor Pedrosa, INSTICC, Portugal

CD-ROM Production

Paulo Brito, INSTICC, Portugal

Webdesigner and Graphics Production

Marina Carvalho, INSTICC, Portugal

Secretariat and Webmaster

Marina Carvalho, INSTICC, Portugal

PROGRAM COMMITTEE

Eugenio Aguirre, University of Granada, Spain

Arturo Hernandez Aguirre, Centre for Research in Mathematics, Mexico

Frank Allgower, University of Stuttgart, Germany

Fouad AL-Sunni, KFUPM, Saudi Arabia

Bala Amavasai, Sheffield Hallam University, U.K.

Francesco Amigoni, Politecnico di Milano, Italy

Yacine Amirat, University Paris 12, France

Nicolas Andreff, LASMEA, France

Stefan Andrei, National University of Singapore, Singapore

Plamen Angelov, Lancaster University, U.K.

Luis Antunes, GUESS/Universidade de Lisboa, Portugal

Peter Arato, Budapest University of Technology and Economics, Hungary

Helder Araújo, University of Coimbra, Portugal

Gustavo Arroyo-Figueroa, Instituto de Investigaciones Electricas, Mexico

Marco Antonio Arteaga, Universidad Nacional Autonoma de Mexico, Mexico

Vijanth Sagayan Asirvadam, University Technology Petronas, Malaysia

Nikos Aspragathos, University of Patras, Greece

Robert Babuska, TU Delft, The Netherlands

Ruth Bars, Budapest University of Technology and Economics, Hungary

Karsten Berns, University Kaiserslautern, Germany

Robert Bicker, University of Newcastle upon Tyne, U.K.

Stjepan Bogdan, University of Zagreb, Faculty of EE&C, Croatia

Patrick Boucher, SUPELEC, France

Alan Bowling, University of Notre Dame, U.S.A.

Edmund Burke, University of Nottingham, U.K.

Kevin Burn, University of Sunderland, U.K.

Clifford Burrows, Innovative Manufacturing Research Centre, U.K.

Luis M. Camarinha-Matos, New University of Lisbon, Portugal

Marco Campi, University of Brescia, Italy

Marc Carreras, University of Girona, Spain

Jorge Martins de Carvalho, FEUP, Portugal

Alicia Casals, Technical University of Catalonia, Spain

Alessandro Casavola, University of Calabria, Italy

Christos Cassandras, Boston University, U.S.A.

Riccardo Cassinis, University of Brescia, Italy

Raja Chatila, LAAS-CNRS, France

Tongwen Chen, University of Alberta, Canada

YangQuan Chen, Utah State University, U.S.A.

Albert M. K. Cheng, University of Houston, U.S.A.

Graziano Chesi, University of Hong Kong, China

Sung-Bae Cho, Yonsei University, Korea

Ryszard S. Choras, University of Technology & Agriculture, Poland

Carlos Coello Coello, CINVESTAV-IPN, Mexico

Patrizio Colaneri, Politecnico di Milano, Italy

António Dourado Correia, University of Coimbra, Portugal

Yechiel Crispin, Embry-Riddle University, U.S.A.

Keshav Dahal, University of Bradford, U.K.

Mariolino De Cecco, DIMS - University of Trento, Italy

Bart De Schutter, Delft University of Technology, The Netherlands

Angel P. del Pobil, Universitat Jaume I, Spain

Guilherme DeSouza, University of Missouri, U.S.A.

Rüdiger Dillmann, University of Karlsruhe, Germany

Feng Ding, Southern Yangtze University, China

Denis Dochain, Université Catholique de Louvain, Belgium

Tony Dodd, The University of Sheffield, U.K.

Alexandre Dolgui, Ecole des Mines de Saint Etienne, France

PROGRAM COMMITTEE (CONT.)

Marco Dorigo, Université Libre de Bruxelles, Belgium

Petr Ekel, Pontifical Catholic University of Minas Gerais, Brazil

Heinz-Hermann Erbe, TU Berlin, Germany

Gerardo Espinosa-Perez, Universidad Nacional Autonoma de Mexico, Mexico

Simon Fabri, University of Malta, Malta

Sergej Fatikow, University of Oldenburg, Germany

Jean-Marc Faure, Ecole Normale Superieure de Cachan, France

Jean-Louis Ferrier, Université d'Angers, France

Florin Gheorghe Filip, The Romanian Academy & The National Institute for R&D in Informatics (ICI), Romania

Georg Frey, University of Kaiserslautern, Germany

Manel Frigola, Technical University of Catalonia (UPC), Spain

Colin Fyfe, University of paisley, U.K.

Dragan Gamberger, Rudjer Boskovic Institute, Croatia

Leonardo Garrido, Tecnológico de Monterrey, Mexico

Ryszard Gessing, Silesian University of Technology, Poland

Lazea Gheorghe, Technical University of Cluj-Napoca, Romania

Maria Gini, University of Minnesota, U.S.A.

Alessandro Giua, University of Cagliari, Italy

Luis Gomes, Universidade Nova de Lisboa, Portugal

John Gray, University of Salford, U.K.

Dongbing Gu, University of Essex, U.K.

Jason Gu, Dalhousie University, Canada

José J. Guerrero, Universidad de Zaragoza, Spain

Jatinder (Jeet) Gupta, University of Alabama in Huntsville, U.S.A.

Thomas Gustafsson, Luleå University of Technology, Sweden

Maki K. Habib, Saga University, Japan

Hani Hagras, University of Essex, U.K.

Wolfgang Halang, Fernuniversitaet, Germany

J. Hallam, University of Southern Denmark, Denmark

Riad Hammoud, Delphi Electronics & Safety, U.S.A.

Uwe D. Hanebeck, Institute of Computer Science and Engineering, Germany

John Harris, University of Florida, U.S.A.

Robert Harrison, The University of Sheffield, U.K.

Vincent Hayward, McGill Univ., Canada

Dominik Henrich, University of Bayreuth, Germany

Francisco Herrera, University of Granada, Spain

Victor Hinostroza, University of Ciudad Juarez, Mexico

Weng Ho, National University of Singapore, Singapore

Wladyslaw Homenda, Warsaw University of Technology, Poland

Alamgir Hossain, Bradford University, U.K.

Dean Hougen, University of Oklahoma, U.S.A.

Amir Hussain, University of Stirling, U.K.

Seth Hutchinson, University of Illinois, U.S.A.

Atsushi Imiya, IMIT Chiba Uni, Japan

Sirkka-Liisa Jämsä-Jounela, Helsinki University of Technology, Finland

Ray Jarvis, Monash University, Australia

Odest Jenkins, Brown University, U.S.A.

Ping Jiang, The University of Bradford, U.K.

Ivan Kalaykov, Örebro University, Sweden

Dimitrios Karras, Chalkis Institute of Technology, Greece

Dusko Katic, Mihailo Pupin Institute, Serbia

Graham Kendall, The University of Nottingham, U.K.

Uwe Kiencke, University of Karlsruhe (TH), Germany

Jozef Korbicz, University of Zielona Gora, Poland

Israel Koren, University of Massachusetts, U.S.A.

PROGRAM COMMITTEE (CONT.)

Bart Kosko, University of Southern California, U.S.A.

George L. Kovács, Hungarian Academy of Sciences, Hungary

Krzysztof Kozlowski, Poznan University of Technology, Poland

Gerhard Kraetzschmar, Fraunhofer Institute for Autonomous Intelligent Systems, Germany

Cecilia Laschi, Scuola Superiore Sant'Anna, Italy

Loo Hay Lee, National University of Singapore, Singapore

Soo-Young Lee, KAIST, Korea

Graham Leedham, University of New South Wales (Asia), Singapore

Cees van Leeuwen, RIKEN BSI, Japan

Kauko Leiviskä, University of Oulu, Finland

Kang Li, Queen's University Belfast, U.K.

Yangmin Li, University of Macau, China

Zongli Lin, University of Virginia, U.S.A.

Cheng-Yuan Liou, National Taiwan University, Taiwan

Vincenzo Lippiello, Università Federico II di Napoli, Italy

Honghai Liu, University of Portsmouth, U.K.

Luís Seabra Lopes, Universidade de Aveiro, Portugal

Brian Lovell, The University of Queensland, Australia

Peter Luh, University of Connecticut, U.S.A.

Anthony Maciejewski, Colorado State University, U.S.A.

N. P. Mahalik, Gwangju Institute of Science and Technology, Korea

Bruno Maione, Politecnico di Bari, Italy

Frederic Maire, Queensland University of Technology, Australia

Om Malik, University of Calgary, Canada

Danilo Mandic, Imperial College, U.K.

Jacek Mandziuk, Warsaw University of Technology, Poland

Hervé Marchand, INRIA, France

Philippe Martinet, LASMEA, France

Aleix Martinez, Ohio State University, U.S.A.

Aníbal Matos, Faculdade de Engenharia da Universidade do Porto (FEUP), Portugal

Rene V. Mayorga, University of Regina, Canada

Barry McCollum, Queen's University Belfast, U.K.

Ken McGarry, University of Sunderland, U.K.

Gerard McKee, The University of Reading, U.K.

Seán McLoone, National University of Ireland (NUI), Maynooth, Ireland

Basil Mertzios, (1)Thessaloniki Institute of Technology, (2) Democritus University, Greece

José Mireles Jr., Universidad Autonoma de Ciudad Juarez, Mexico

Sushmita Mitra, Indian Statistical Institute, India

Vladimir Mostyn, VSB - Technical University of Ostrava, Czech Republic

Rafael Muñoz-Salinas, University of Cordoba, Spain

Kenneth Muske, Villanova University, U.S.A.

Ould Khessal Nadir, Okanagan College, Canada

Fazel Naghdy, University of Wollongong, Australia

Tomoharu Nakashima, Osaka Prefecture University, Japan

Andreas Nearchou, University of Patras, Greece

Luciana Porcher Nedel, Universidade Federal do Rio Grande do Sul (UFRGS), Brazil

Sergiu Nedevschi, Technical University of Cluj-Napoca, Romania

Maria Neves, Instituto Superior de Engenharia do Porto, Portugal

Hendrik Nijmeijer, Eindhoven University of Technology, The Netherlands

Juan A. Nolazco-Flores, ITESM, Campus Monterrey, Mexico

Urbano Nunes, University of Coimbra, Portugal

Gustavo Olague, CICESE, Mexico

PROGRAM COMMITTEE (CONT.)

José Valente de Oliveira, Universidade do Algarve, Portugal

Andrzej Ordys, Kingston University in London, Faculty of Engineering, U.K.

Djamila Ouelhadj, University of Nottingham, ASAP GROUP (Automated Scheduling, Optimisation and Planning), U.K.

Manuel Ortigueira, Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa, Portugal

Christos Panayiotou, University of Cyprus, Cyprus

Evangelos Papadopoulos, NTUA, Greece

Panos Pardalos, University of Florida, U.S.A.

Michel Parent, INRIA, France

Thomas Parisini, University of Trieste, Italy

Igor Paromtchik, RIKEN, Japan

Gabriella Pasi, Università degli Studi di Milano Bicocca, Italy

Witold Pedrycz, University of Alberta, Canada

Carlos Eduardo Pereira, Federal University of Rio Grande do Sul - UFRGS, Brazil

Maria Petrou, Imperial College, U.K.

J. Norberto Pires, University of Coimbra, Portugal

Marios Polycarpou, University of Cyprus, Cyprus

Marie-Noëlle Pons, CNRS, France

Libor Preucil, Czech Technical University in Prague, Czech Republic

Joseba Quevedo, Technical University of Catalonia, Spain

Robert Reynolds, Wayne State University, U.S.A.

A. Fernando Ribeiro, Universidade do Minho, Portugal

Bernardete Ribeiro, University of Coimbra, Portugal

Robert Richardson, University of Manchester, U.K.

John Ringwood, National University of Ireland (NUI), Maynooth, Ireland

Rodney Roberts, Florida State University, U.S.A.

Kurt Rohloff, BBN Technologies, U.S.A.

Juha Röning, University of Oulu, Finland

Agostinho Rosa, IST, Portugal

Hubert Roth, University Siegen, Germany

António Ruano, CSI, Portugal

Carlos Sagüés, University of Zaragoza, Spain

Mehmet Sahinkaya, University of Bath, U.K.

Antonio Sala, Universidad Politecnica de Valencia, Spain

Abdel-Badeeh Salem, Ain Shams University, Egypt

Ricardo Sanz, Universidad Politécnica de Madrid, Spain

Medha Sarkar, Middle Tennessee State University, U.S.A.

Nilanjan Sarkar, Vanderbilt University, U.S.A.

Jurek Sasiadek, Carleton University, Canada

Daniel Sbarbaro, Universidad de Concepcion, Chile

Carsten Scherer, Delft University of Technology, The Netherlands

Carla Seatzu, University of Cagliari, Italy

Klaus Schilling, University Würzburg, Germany

Yang Shi, University of Saskatchewan, Canada

Michael Short, University of Leicester, U.K.

Chi-Ren Shyu, University of Missouri-Columbia, U.S.A.

Bruno Siciliano, Università di Napoli Federico II, Italy

João Silva Sequeira, Instituto Superior Técnico, Portugal

Silvio Simani, University of Ferrara, Italy

Amanda Sharkey, University of Sheffield, U.K.

Michael Small, Hong Kong Polytechnic University, China

Burkhard Stadlmann, University of Applied Sciences Wels, Austria

Tarasiewicz Stanislaw, Université Laval, Canada

Aleksandar Stankovic, Northeastern University, U.S.A.

Raúl Suárez, Universitat Politecnica de Catalunya (UPC), Spain

PROGRAM COMMITTEE (CONT.)

Ryszard Tadeusiewicz, AGH University of Science and Technology, Poland

Tianhao Tang, Shanghai Maritime University, China

Adriana Tapus, University of Southern California, U.S.A.

József K. Tar, Budapest Tech Polytechnical Institution, Hungary

Daniel Thalmann, EPFL, Switzerland

Gui Yun Tian, University of Huddersfield, U.K.

Antonios Tsourdos, Cranfield University, U.K.

Nikos Tsourveloudis, Technical University of Crete, Greece

Ivan Tyukin, RIKEN Brain Science Institute, Japan

Masaru Uchiyama, Tohoku University, Japan

Nicolas Kemper Valverde, Universidad Nacional Autónoma de México, Mexico

Marc Van Hulle, K. U. Leuven, Belgium

Annamaria R. Varkonyi-Koczy, Budapest University of Technology and Economics, Hungary

Luigi Villani, Università di Napoli Federico II, Italy

Markus Vincze, Technische Universität Wien, Austria

Bernardo Wagner, University of Hannover, Germany

Axel Walthelm, sepp.med gmbh, Germany

Lipo Wang, Nanyang Technological University, Singapore

Alfredo Weitzenfeld, ITAM, Mexico

Dirk Wollherr, Technische Universität München, Germany

Sangchul Won, Pohang University of Science and Technology, Korea

Kainam Thomas Wong, The Hong Kong Polytechnic University, China

Jeremy Wyatt, University of Birmingham, U.K.

Alex Yakovlev, University of Newcastle, U.K.

Hujun Yin, University of Manchester, U.K.

Xinghuo Yu, Royal Melbourne Institute of Techology, Australia

Du Zhang, California State University, U.S.A.

Janusz Zalewski, Florida Gulf Coast University, U.S.A.

Marek Zaremba, Universite du Quebec, Canada

Dayong Zhou, University of Oklahoma, U.S.A.

Argyrios Zolotas, Loughborough University, U.K.

Albert Zomaya, The University of Sydney, Austrália

AUXILIARY REVIEWERS

Rudwan Abdullah, University of Stirling, U.K.

Luca Baglivo, University of Padova, Italy

Prasanna Balaprakash, IRIDIA, Université Libre de Bruxelles, Belgium

João Balsa, Universidade de Lisboa, Portugal

Alejandra Barrera, ITAM, Mexico

Frederik Beutler, Intelligent Sensor-Actuator-Systems Laboratory - Universität Karlsruhe (TH), Germany

Alecio Binotto, CETA SENAI-RS, Brazil

Nizar Bouguila, Concordia University, Canada

Dietrich Brunn, Intelligent Sensor-Actuator-Systems Laboratory - Universität Karlsruhe (TH), Germany

Maria Paola Cabasino, Dip.to Ingegneria Elettrica ed Elettronica Universita' di Cagliari, Italy

Joao Paulo Caldeira, EST-IPS, Portugal

Aneesh Chauhan, Universidade de Aveiro, Portugal

Paulo Gomes da Costa, FEUP, Portugal

Xevi Cufi, University of Girona, Spain

Sérgio Reis Cunha, FEUP, Portugal

Paul Dawson, Boise State University, U.S.A.

Mahmood Elfandi, Elfateh University, Libya

AUXILIARY REVIEWERS (CONT.)

Michele Folgheraiter, Politecnico di Milano, Italy

Diamantino Freitas, FEUP, Portugal

Reinhard Gahleitner, University of Applied Sciences Wels, Austria

Nils Hagge, Leibniz Universität Hannover, Institute for Systems Engineering, Germany

Onur Hamsici, The Ohio State Univeristy, U.S.A.

Renato Ventura Bayan Henriques, UFRGS, Brazil

Matthias Hentschel, Leibniz Universität Hannover, Institute for Systems Engineering, Germany

Marco Huber, Intelligent Sensor-Actuator-Systems Laboratory - Universität Karlsruhe (TH), Germany

Markus Kemper, University of Oldenbvurg, Germany

Vesa Klumpp, Intelligent Sensor-Actuator-Systems Laboratory - Universität Karlsruhe (TH), Germany

Daniel Lecking, Leibniz Universität Hannover, Institute for Systems Engineering, Germany

Gonzalo Lopez-Nicolas, University of Zaragoza, Spain

Cristian Mahulea, University of Zaragoza, Spain

Cristian Mahulea, Dep.to Informática e Ingeneiría de Sistemas Centro Politécnico Superior, Spain

Nikolay Manyakov, K. U. Leuven, Belgium

Antonio Muñoz, University of Zaragoza, Spain

Ana C. Murillo, University of Zaragoza, Spain

Andreas Neacrhou, University of Patras, Greece

Marco Montes de Oca, IRIDIA, Université Libre de Bruxelles, Belgium

Sorin Olaru, Supelec, France

Karl Pauwels, K. U. Leuven, Belgium

Luis Puig, University of Zaragoza, Spain

Ana Respício, Universidade de Lisboa, Portugal

Pere Ridao, University of Girona, Spain

Kathrin Roberts, Intelligent Sensor-Actuator-Systems Laboratory - Universität Karlsruhe (TH), Germany

Paulo Lopes dos Santos, FEUP, Portugal

Felix Sawo, Intelligent Sensor-Actuator-Systems Laboratory - Universität Karlsruhe (TH), Germany

Frederico Schaf, UFRGS, Brazil

Oliver Schrempf, Intelligent Sensor-Actuator-Systems Laboratory - Universität Karlsruhe (TH), Germany

Torsten Sievers, University of Oldenbvurg, Germany

Razvan Solea, Institute of Systems and Robotics, University of Coimbra, Portugal

Wolfgang Steiner, University of Applied Sciences Wels, Austria

Christian Stolle, University of Oldenbvurg, Germany

Alina Tarau, Delft University of Technology, The Netherlands

Rui Tavares, University of Evora, Portugal

Paulo Trigo, ISEL, Portugal

Haralambos Valsamos, University of Patras, Greece

José Luis Villarroel, University of Zaragoza, Spain

Yunhua Wang, University of Oklahoma, U.S.A.

Florian Weißel, Intelligent Sensor-Actuator-Systems Laboratory - Universität Karlsruhe (TH), Germany

Jiann-Ming Wu, National Dong Hwa University, Taiwan

Oliver Wulf, Leibniz Universität Hannover, Institute for Systems Engineering, Germany

Ali Zayed, Seventh of April University, Libya

Yan Zhai, University of Oklahoma, U.S.A.

SELECTED PAPERS BOOK

A number of selected papers presented at ICINCO 2007 will be published by Springer, in a book entitled Informatics in Control, Automation and Robotics IV. This selection will be done by the conference co-chairs and program co-chairs, among the papers actually presented at the conference, based on a rigorous review by the ICINCO 2007 program committee members.

FOREWORD

Welcome to the 4th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2007) held at the University of Angers. The ICINCO Conference Series has now consolidated as a major forum to debate technical and scientific advances presented by researchers and developers both from academia and industry, working in areas related to Control, Automation and Robotics that require Information Technology support.

In this year Conference Program we have included oral presentations (full papers and short papers) as well as posters, organized in three simultaneous tracks: "Intelligent Control Systems and Optimization", "Robotics and Automation" and "Systems Modeling, Signal Processing and Control". Furthermore, ICINCO 2007 includes 2 satellite workshops and 3 plenary keynote lectures, given by internationally recognized researchers

The two satellite workshops that are held in conjunction with ICINCO 2007 are: Third International Workshop on Multi-Agent Robotic Systems (MARS 2007) and Third International Workshop on Artificial Neural Networks and Intelligent Information Processing (ANNIIP 2007).

As additional points of interest, it is worth mentioning that the Conference Program includes a plenary panel subject to the theme "Practical Applications of Intelligent Control and Robotics" and 3 Special Sessions focused on very specialized topics.

ICINCO has received 435 paper submissions, not including workshops, from more than 50 countries, in all continents. To evaluate each submission, a double blind paper review was performed by the program committee, whose members are researchers in one of ICINCO main topic areas. Finally, only 263 papers are published in these proceedings and presented at the conference; of these, 195 papers were selected for oral presentation (52 full papers and 143 short papers) and 68 papers were selected for poster presentation. The global acceptance ratio was 60,4% and the full paper acceptance ratio was 11,9%. After the conference, some authors will be invited to publish extended versions of their papers in a journal and a short list of about thirty papers will be included in a book that will be published by Springer with the best papers of ICINCO 2007.

In order to promote the development of research and professional networks the conference includes in its social program a Town Hall Reception in the evening of May 9 (Wednesday) and a Conference and Workshops Social Event & Banquet in the evening of May 10 (Thursday).

We would like to express our thanks to all participants. First of all to the authors, whose quality work is the essence of this conference. Next, to all the members of the Program Committee and

reviewers, who helped us with their expertise and valuable time. We would also like to deeply thank the invited speakers for their excellent contribution in sharing their knowledge and vision. Finally, a word of appreciation for the hard work of the secretariat; organizing a conference of this level is a task that can only be achieved by the collaborative effort of a dedicated and highly capable team.

Commitment to high quality standards is a major aspect of ICINCO that we will strive to maintain and reinforce next year, including the quality of the keynote lectures, of the workshops, of the papers, of the organization and other aspects of the conference. We look forward to seeing more results of R&D work in Informatics, Control, Automation and Robotics at ICINCO 2008, next May, at the Hotel Tivoli Ocean Park, Funchal, Madeira, Portugal.

Janan Zaytoon

CReSTIC, URCA, France

Juan Andrade Cetto

Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Spain

Jean-Louis Ferrier

Université d'Angers, France

Joaquim Filipe

Polytechnic Institute of Setúbal / INSTICC, Portugal

CONTENTS

SHORT PAPERS

POSTERS

SPECIAL SESSION ON "FROM PLANNING TO CONTROL OF MANUFACTURING SYSTEMS"

INVITED SPEAKERS

KEYNOTE LECTURES

REAL TIME DIAGNOSTICS, PROGNOSTICS, & PROCESS MODELING

Dimitar Filev
The Ford Motor Company
U.S.A.

Abstract: Practical and theoretical problems related to the design of real time diagnostics, prognostics, & process modeling systems are discussed. Major algorithms for autonomous monitoring of machine health in industrial networks are proposed and relevant architectures for incorporation of intelligent prognostics within plant floor information systems are reviewed. Special attention is given to the practical realization of real time structure and parameter learning algorithms. Links between statistical process control and real time modeling based on the evolving system paradigm are analyzed relative to the design of soft sensing algorithms. Examples and case studies of industrial implementation of aforementioned concepts are presented.

BRIEF BIOGRAPHY

Dr. Dimitar P. Filev is a Senior Technical Leader, Intelligent Control & Information Systems with Ford Motor Company specializing in industrial intelligent systems and technologies for control, diagnostics and decision making. He is conducting research in systems theory and applications, modeling of complex systems, intelligent modeling and control and he has published 3 books, and over 160 articles in refereed journals and conference proceedings. He holds15 granted U.S. patents and numerous foreign patents in the area of industrial intelligent systems Dr. Filev is a recipient of the '95 Award for Excellence of MCB University Press and was awarded 4 times with the Henry Ford Technology Award for development and implementation of advanced intelligent control technologies. He is Associate Editor of Int. J. of General Systems and Int. J. of Approximate Reasoning. He is a member of the Board of Governors of the IEEE Systems, Man & Cybernetics Society and President of the North American Fuzzy Information Processing Society (NAFIPS). Dr. Filev received his PhD. degree in Electrical Engineering from the Czech Technical University in Prague in 1979.

SYNCHRONIZATION OF MULTI-AGENT SYSTEMS

Mark W. Spong
Donald Biggar Willett Professor of Engineering
Professor of Electrical and Computer Engineering
Coordinated Science Laboratory
University of Illinois at Urbana-Champaign
U.S.A.

Abstract: There is currently great interest in the control of multi-agent networked systems. Applications include mobile sensor networks, teleoperation, synchronization of oscillators, UAV's and coordination of multiple robots. In this talk we consider the output synchronization of networked dynamic agents using passivity theory and considering the graph topology of the inter-agent communication. We provide a coupling control law that results in output synchronization and we discuss the extension to state synchronization in addition to output synchronization. We also consider the extension of these ideas to systems with time delay in communication among agents and obtain results showing synchronization for arbitrary time delay. We will present applications of our results in synchronization of Kuramoto oscillators and in bilateral teleoperators.

BRIEF BIOGRAPHY

Mark W. Spong received the B.A. degree, magna cum laude and Phi Beta Kappa, in mathematics and physics from Hiram College, Hiram, Ohio in 1975, the M.S. degree in mathematics from New Mexico State University in 1977, and the M.S. and D.Sc. degrees in systems science and mathematics in 1979 and 1981, respectively, from Washington University in St. Louis. Since 1984 he has been at the University of Illinois at Urbana-Champaign where he is currently a Donald Biggar Willett Distinguished Professor of Engineering, Professor of Electrical and Computer Engineering, and Director of the Center for Autonomous Engineering Systems and Robotics. Dr. Spong is Past President of the IEEE Control Systems Society and a Fellow of the IEEE. Dr. Spong's main research interests are in robotics, mechatronics, and nonlinear control theory. He has published more than 200 technical articles in control and robotics and is co-author of four books. His recent awards include the Senior U.S. Scientist Research Award from the Alexander von Humboldt Foundation, the Distinguished Member Award from the IEEE Control Systems Society, the John R. Ragazzini and O. Hugo Schuck Awards from the American Automatic Control Council, and the IEEE Third Millennium Medal.

TOWARD HUMAN-MACHINE COOPERATION

Patrick Millot
Laboratoire d'Automatique, de Mécanique et d'Informatique Industrielle et Humaine
Université de Valenciennes
France

Abstract: In human machine systems human activities are mainly oriented toward decision-making: monitoring and fault detection, fault anticipation, diagnosis and prognosis, and fault prevention and recovery. The objectives combine the human-machine system performances (production quantity and quality) as well as the global system safety. In this context human operators may have a double role: (1) a negative role as they may perform unsafe or erroneous actions on the process, (2) a positive role as they can detect, prevent or recover an unsafe process behavior due to an other operator or to automated decision makers. Two approachs to these questions are combined in a pluridisciplinary research way : (1) human engineering which aims at designing dedicated assistance tools for human operators and at integrating them into human activities through a human machine cooperation, (2) cognitive psychology and ergonomics analysing the human activities, the need for such tools and their use. This paper focuses on the concept of cooperation and proposes a framework for implementation. Examples in Air Traffic Control and in Telecommunication networks illustrate these concepts.

BRIEF BIOGRAPHY

Born in 53 he received a PhD in Automatic Control (79) an is Docteur d'Etat es Sciences (87). He is full Professor at the University of Valenciennes since 89. He conducts research on Automation Sciences, Artificial Intelligence, Supervisory Control, Human Machine Systems, Human Reliability with applications to production telecommunication and transport systems (Air Traffic Control, Car Traffic, Trains Metro.). His scientific production covers about 175 publications, collective books, conference proceedings. Research Director of 35 PhD students and 9 HDR since 89, reviewer of 50 PhD Thesis and 9 HDR from other universities. Head of the research group "Human Machine Systems" in LAMIH since 87 till 04 (25 researchers). Vice-head then head of LAMIH between 96 and 05 (222 researchers and engineers). Vice Chairman of the University of Valenciennes since October 05 in charge of research.

Scientific head or Member of the scientific board or Manager of several regional research groups on Supervisory Control (GRAISYHM 96-02) on Transport System Safety (GRRT since 87,pôle ST2 since 01 with 80 researchers of 10 labs). Member of the French Council of the Universities (96-03), member of the scientific board of the french national research group in Automation Sciences supported by CNRS (96-01). Partner of several European projects and netwoks (HCM networks 93-96, 2 projects since 02 on Urban Guided Transport Management Systems and the Network of Excellence EURNEX since 04). Member of the IFAC Technical Committee 4.5 Human Machine Systems since 00. IPC member of several International Conferences and Journals.

INTELLIGENT CONTROL SYSTEMS AND OPTIMIZATION

FULL PAPERS

BINARY OPTIMIZATION: A RELATION BETWEEN THE DEPTH OF A LOCAL MINIMUM AND THE PROBABILITY OF ITS DETECTION

B. V. Kryzhanovsky, V. M. Kryzhanovsky and A. L. Mikaelian
Center of Optical Neural Technologies, SR Institute of System Analisys RAS
44/2 Vavilov Str, Moscow 119333, Russia
kryzhanov@mail.ru, iont@iont.ru

Keywords: Binary optimization, neural networks, local minimum.

Abstract: The standard method in optimization problems consists in a random search of the global minimum: a neuron network relaxes in the nearest local minimum from some randomly chosen initial configuration. This procedure is to be repeated many times in order to find as deep an energy minimum as possible. However the question about the reasonable number of such random starts and whether the result of the search can be treated as successful remains always open. In this paper by analyzing the generalized Hopfield model we obtain expressions describing the relationship between the depth of a local minimum and the size of the basin of attraction. Based on this, we present the probability of finding a local minimum as a function of the depth of the minimum. Such a relation can be used in optimization applications: it allows one, basing on a series of already found minima, to estimate the probability of finding a deeper minimum, and to decide in favor of or against further running the program. The theory is in a good agreement with experimental results.

1 INTRODUCTION

Usually a neural system of associative memory is considered as a system performing a recognition or retrieval task. However it can also be considered as a system that solves an optimization problem: the network is expected to find a configuration minimizes an energy function (Hopfield,1982). This property of a neural network can be used to solve different *NP*-complete problems. A conventional approach consists in finding such an architecture and parameters of a neural network, at which the objective function or cost function represents the neural network energy. Successful application of neural networks to the traveling salesman problem (Hopfield and Tank, 1985) had initiated extensive investigations of neural network approaches for the graph bipartition problem (Fu and Anderson, 1986), neural network optimization of the image processing (Poggio and Girosi, 1990) and many other applications. This subfield of the neural network theory is developing rapidly at the moment (Smith, 1999), (Hartmann and Rieger, 2004), (Huajin Tang et al, 2004), (Kwok and Smith, 2004), (Salcedo-Sanz et al, 2004), (Wang et al, 2004, 2006).

The aforementioned investigations have the same common feature: the overwhelming majority of neural network optimization algorithms contain the Hopfield model in their core, and the optimization process is reduced to finding the global minimum of some quadratic functional (the energy) constructed on a given $N \times N$ matrix in an N-dimensional configuration space (Joya, 2002), (Kryzhanovsky et al, 2005). The standard neural network approach to such a problem consists in a random search of an optimal solution. The procedure consists of two stages. During the first stage the neural network is initialized at random, and during the second stage the neural network relaxes into one of the possible stable states, i.e. it optimizes the energy value. Since the sought result is unknown and the search is done at random, the neural network is to be initialized many times in order to find as deep an energy minimum as possible. But the question about the reasonable number of such random starts and whether the result of the search can be regarded as successful always remains open.

In this paper we have obtained expressions that have demonstrated the relationship between the depth of a local minimum of energy and the size of

the basin of attraction (Kryzhanovsky et al, 2006). Based on this expressions, we presented the probability of finding a local minimum as a function of the depth of the minimum. Such a relation can be used in optimization applications: it allows one, based on a series of already found minima, to estimate the probability of finding a deeper minimum, and to decide in favor of or against further running of the program. Our expressions are obtained from the analysis of generalized Hopfield model, namely, of a neural network with Hebbian matrix. They are however valid for any matrices, because any kind of matrix can be represented as a Hebbian one, constructed on arbitrary number of patterns. A good agreement between our theory and experiment is obtained.

2 DESCRIPTION OF THE MODEL

Let us consider Hopfield model, i.e a system of N Ising spins-neurons $s_i = \pm 1$, $i = \overline{1,N}$. A state of such a neural network can be characterized by a configuration $\boldsymbol{S} = (s_1, s_2, ..., s_N)$. Here we consider a generalized model, in which the connection matrix:

$$T_{ij} = \sum_{m=1}^{M} r_m s_i^{(m)} s_j^{(m)}, \qquad \sum r_m^2 = 1 \tag{1}$$

is constructed following Hebbian rule on M binary N-dimensional patterns $\boldsymbol{S}_m = (s_1^{(m)}, s_2^{(m)}, ..., s_N^{(m)})$, $m = \overline{1,M}$. The diagonal matrix elements are equal to zero ($T_{ii} = 0$). The generalization consists in the fact, that each pattern $\boldsymbol{S}_m$ is added to the matrix T_{ij} with its statistical weight r_m. We normalize the statistical weights to simplify the expressions without loss of generality. Such a slight modification of the model turns out to be essential, since in contrast to the conventional model it allows one to describe a neural network with a non-degenerate spectrum of minima.

The energy of the neural network is given by the expression:

$$E = -\tfrac{1}{2} \sum_{i,j=1}^{N} s_i T_{ij} s_j \tag{2}$$

and its (asynchronous) dynamics consist in the following. Let $\boldsymbol{S}$ be an initial state of the network. Then the local field $h_i = -\partial E / \partial s_i$, which acts on a randomly chosen i-th spin, can be calculated, and the energy of the spin in this field $\varepsilon_i = -s_i h_i$ can be determined. If the direction of the spin coincides with the direction of the local field ($\varepsilon_i < 0$), then its state is stable, and in the subsequent moment ($t+1$) its state will undergo no changes. In the opposite case ($\varepsilon_i > 0$) the state of the spin is unstable and it flips along the direction of the local field, so that $s_i(t+1) = -s_i(t)$ with the energy $\varepsilon_i(t+1) < 0$. Such a procedure is to be sequentially applied to all the spins of the neural network. Each spin flip is accompanied by a lowering of the neural network energy. It means that after a finite number of steps the network will relax to a stable state, which corresponds to a local energy minimum.

3 BASIN OF ATTRACTION

Let us examine at which conditions the pattern $\boldsymbol{S}_m$ embeded in the matrix (1) will be a stable point, at which the energy E of the system reaches its (local) minimum E_m. In order to obtain correct estimates we consider the asymptotic limit $N \to \infty$. We determine the basin of attraction of a pattern $\boldsymbol{S}_m$ as a set of the points of N-dimensional space, from which the neural network relaxes into the configuration $\boldsymbol{S}_m$. Let us try to estimate the size of this basin. Let the initial state of the network $\boldsymbol{S}$ be located in a vicinity of the pattern $\boldsymbol{S}_m$. Then the probability of the network convergation into the point $\boldsymbol{S}_m$ is given by the expression:

$$\mathbf{Pr} = \left(\frac{1 + \mathbf{erf}\, \gamma}{2} \right)^N \tag{3}$$

where $\mathbf{erf}\, \gamma$ is the error function of the variable γ:

$$\gamma = \frac{r_m \sqrt{N}}{\sqrt{2(1 - r_m^2)}} \left(1 - \frac{2n}{N} \right) \tag{4}$$

and n is Hemming distance between $\boldsymbol{S}_\mathrm{m}$ and $\boldsymbol{S}$. The expression (3) can be obtained with the help of the methods of probability theory, repeating the well-known calculation (Perez-Vincente, 1989) for conventional Hopfield model.

It follows from (3) that the basin of attraction is determined as the set of the points of the configuration space close to $\boldsymbol{S}_m$, for which $n \le n_m$:

$$n_m = \frac{N}{2}\left(1 - \frac{r_0\sqrt{1-r_m^2}}{r_m\sqrt{1-r_0^2}}\right) \qquad (5)$$

where

$$r_0 = \sqrt{2\ln N / N} \qquad (6)$$

Indeed, if $n \le n_m$ we have $\mathbf{Pr} \to 1$ for $N \to \infty$, i.e. the probability of the convergence to the point $\boldsymbol{S}_m$ asymptotically tends to 1. In the opposite case ($n > n_m$) we have $\mathbf{Pr} \to 0$. It means that the quantity n_m can be considered as the radius of the basin of attraction of the local minimum E_m.

It follows from (5) that the radius of basin of attraction tends to zero when $r_m \to r_0$ (Fig.1). It means that the patterns added to the matrix (1), whose statistical weight is smaller than r_0, simply do not form local minima. Local minima exist only in those points $\boldsymbol{S}_m$, whose statistical weight is relatively large: $r_m > r_0$.

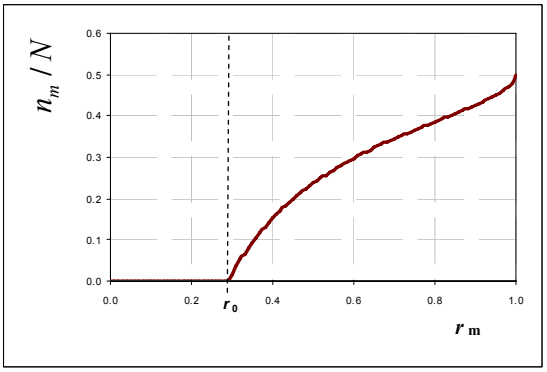


Figure 1: A typical dependence of the width of the basin of attraction n_m on the statistical weight of the pattern r_m. A local minimum exists only for those patterns, whose statistical weight is greater than r_0. For $r_m \to r_0$ the size of the basin of attraction tends to zero, i.e. the patters whose statistical weight $r_m \le r_0$ do not form local minima.

4 DEPTH OF LOCAL MINIMUM

From analysis of Eq. (2) it follows that the energy of a local minimum E_m can be represented in the form:

$$E_m = -r_m N^2 \qquad (7)$$

with the accuracy up to an insignificant fluctuation of the order of

$$\sigma_m = N\sqrt{1-r_m^2} \qquad (8)$$

Then, taking into account Eqs. (5) and (7), one can easily obtain the following expression:

$$E_m = E_{\min}\frac{1}{\sqrt{(1-2n_m/N)^2 + E_{\min}^2/E_{\max}^2}} \qquad (9)$$

where

$$E_{\min} = -N\sqrt{2N\ln N}\ , \quad E_{\max} = -\left(\sum_{m=1}^{M} E_m^2\right)^{1/2} \qquad (10)$$

which yield a relationship between the depth of the local minimum and the size of its basin of attraction. One can see that the wider the basin of attraction, the deeper the local minimum and vice versa: the deeper the minimum, the wider its basin of attraction (see Fig.2).

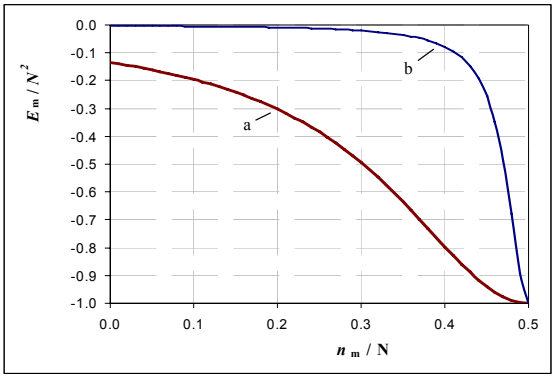


Figure 2: The dependence of the energy of a local minimum on the size of the basin of attraction: a) N=50; b) N=5000.

We have introduced here also a constant E_{max}, which we make use of in what follows. It denotes the maximal possible depth of a local minimum. In the adopted normalization, there is no special need to introduce this new notation, since it follows from (7)-(9) that $E_{\max} = -N^2$. However for other normalizations some other dependencies of $E_{\max}$ on N are possible, which can lead to a misunderstanding.

The quantity E_{min} introduced in (10) characterizes simultaneously two parameters of the neural network. First, it determines the half-width of the Lorentzian distribution (9). Second, it follows from (9) that:

$$E_{\max} \le E_m \le E_{\min} \qquad (11)$$

i.e. $E_{\min}$ is the upper boundary of the local minimum spectrum and characterizes the minimal possible depth of the local minimum. These results are in a good agreement with the results of computer experiments aimed to check whether there is a local minimum at the point $\boldsymbol{S}_{\mathrm{m}}$ or not. The results of one of these experiments (N=500, M=25) are shown in Fig.3. One can see a good linear dependence of the energy of the local minimum on the value of the statistical weight of the pattern. Note that the overwhelming number of the experimental points corresponding to the local minima are situated in the right lower quadrant, where $r_m > r_0$ and $E_{\mathrm{m}} < E_{\min}$. One can also see from Fig.3 that, in accordance with (8), the dispersion of the energies of the minima decreases with the increase of the statistical weight.

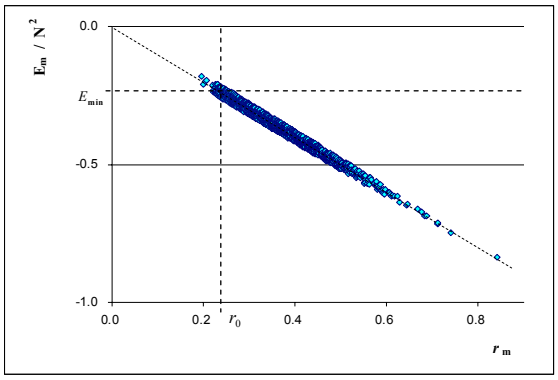


Figure 3: The dependence of the energy E_m of a local minimum on the statistical weight r_m of the pattern.

5 THE PROBABILITY OF FINDING THE MINIMUM

Let us find the probability W of finding a local minimum E_m at a random search. By definition, this probability coincides with the probability for a randomly chosen initial configuration to get to the basin of attraction of the pattern $\boldsymbol{S}_m$. Consequently, the quantity $W = W(n_m)$ is the number of points in a sphere of a radius n_m, reduced to the total number of the points in the N-dimensional space:

$$W = 2^{-N} \sum_{n=1}^{n_m} C_N^n \qquad (12)$$

Equations (5) and (12) define implicitly a connection between the depth of the local minimum and the probability of its finding. Applying asymptotical Stirling expansion to the binomial coefficients and passing from summation to integration one can represent (12) as

$$W = W_0 \, e^{-Nh} \qquad (13)$$

where h is generalized Shannon function

$$h = \frac{n_m}{N} \ln \frac{n_m}{N} + \left(1 - \frac{n_m}{N}\right) \ln\left(1 - \frac{n_m}{N}\right) + \ln 2 \qquad (14)$$

Here W_0 is an insignificant for the further analysis slow function of E_m. It can be obtained from the asymptotic estimate (13) under the condition $n_m >> 1$, and the dependence $W = W(n_m)$ is determined completely by the fast exponent.

It follows from (14) that the probability of finding a local minimum of a small depth ($E_m \sim E_{\min}$) is small and decreases as $W \sim 2^{-N}$. The probability W becomes visibly non-zero only for deep enough minima $|E_m| >> |E_{\min}|$, whose basin of attraction sizes are comparable with $N/2$. Taking into account (9), the expression (14) can be transformed in this case to a dependence $W = W(E_m)$ given by

$$W = W_0 \exp\left[- N E_{\min}^2 \left(\frac{1}{E_m^2} - \frac{1}{E_{\max}^2} \right) \right] \qquad (15)$$

It follows from (14) that the probability to find a minimum increases with the increase of its depth. This dependence"the deeper minimum $\rightarrow$ the larger the basin of attraction $\rightarrow$ the larger the probability to get to this minimum" is confirmed by the results of numerous experiments. In Fig.4 the solid line is computed from Eq. (13), and the points correspond to the experiment (Hebbian matrix with a small loading parameter $M / N \le 0.1$). One can see that a good agreement is achieved first of all for the deepest minima, which correspond to the patterns $\boldsymbol{S}_m$ (the energy interval $E_m \le -0.49 N^2$ in Fig.4). The experimentally found minima of small depth (the points in the region $E_m > -0.44 N^2$) are the so-called "chimeras". In standard Hopfield model ($r_m \equiv 1/\sqrt{M}$) they appear at relatively large loading parameter $M / N > 0.05$. In the more general case, which we consider here, they can appear also earlier. The reasons leading to their appearance are well examined with the help of the methods of statistical physics in (Amit et al, 1985), where it was shown that the chimeras appear as a consequence of

interference of the minima of $\boldsymbol{S}_m$. At a small loading parameter the chimeras are separated from the minima of $\boldsymbol{S}_m$ by an energy gap clearly seen in Fig.4.

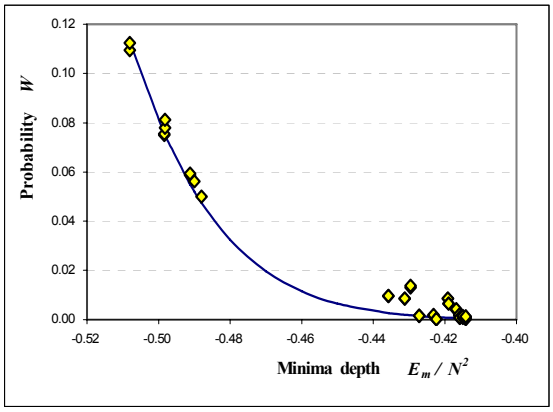


Figure 4: The dependence of the probability W to find a local minimum on its depth E_m: theory - solid line, experiment – points.

6 DISCUSSION

Our analysis shows that the properties of the generalized model are described by three parameters r_0, $E_{\min}$ and $E_{\max}$. The first determines the minimal value of the statistical weight at which the pattern forms a local minimum. The second and third parameters are accordingly the minimal and the maximal depth of the local minima. It is important that these parameters are independent from the number of embeded patterns M.

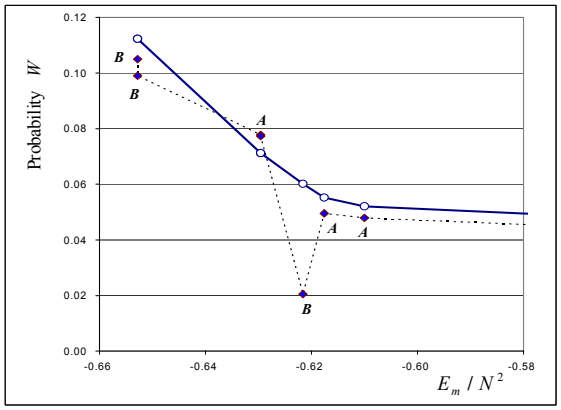


Figure 5: The comparison of the predicted probabilities (solid line) and the experimentally found values (points connected with the dashed line).

Now we are able to formulate a heuristic approach of finding the global minimum of the functional (2) for any given matrix (not necessarily Hebbian one). The idea is to use the expression (15) with unknown parameters W_0, $E_{\min}$ and $E_{\max}$. To do this one starts the procedure of the random search and finds some minima. Using the obtained data, one determines typical values of $E_{\min}$ and $E_{\max}$ and the fitting parameter W_0 for the given matrix. Substituting these values into (15) one can estimate the probability of finding an unknown deeper minimum E_m (if it exists) and decide in favor or against (if the estimate is a pessimistic one) the further running of the program.

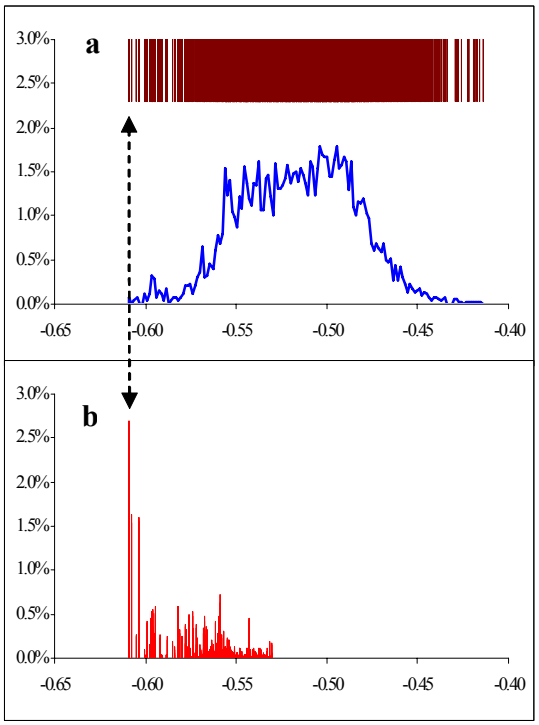


Figure 6: The case of matrix with a quasi-continuous type of spectrum. a) The upper part of the figure shows the spectrum of minima distribution – each vertical line corresponds to a particular minimum. The solid line denotes the spectral density of minima (the number of minima at length ΔE). The Y-axis presents spectral density and the X-axis is the normalized values of energy minima E/N^2. b) Probability of finding a minimum with energy E. The Y-axis is the probability of finding a particular minimum (%) and the X-axis is the normalized values of energy minima.

This approach was tested with Hebbian matrices at relatively large values of the loading parameter ($M/N \geq 0.2 \div 10$). The result of one of the experiments is shown in Fig.5. In this experiment

with the aid of the found minima (the points A) the parameters W_0, $E_{\min}$ and $E_{\max}$ were calculated, and the dependence $W = W(E_m)$ (solid line) was found. After repeating the procedure of the random search over and over again ($\sim 10^5$ random starts) other minima (points B) and the precise probabilities of getting into them were found. One can see that although some dispersion is present, the predicted values in the order of magnitude are in a good agreement with the precise probabilities.

In conclusion we stress once again that any given matrix can be performed in the form of Hebbian matrix (1) constructed on an arbitrary number of patterns (for instance, $M \to \infty$) with arbitrary statistical weights. It means that the dependence "the deeper minimum $\leftrightarrow$ the larger the basin of attraction $\leftrightarrow$ the larger the probability to get to this minimum" as well as all other results obtained in this paper are valid for all kinds of matrices. To prove this dependence, we have generated random matrices, with uniformly distributed elements on [-1,1] segment. The results of a local minima search on one of such matrices are shown in Fig. 6. The value of normalized energy is shown on the X-scale and on the Y-scale the spectral density is noted. As we can see, there are a lot of local minima, and most of them concentrated in central part of spectrum (Fig 6.a). Despite of such a complex view of the spectrum of minima, the deepest minimum is found with maximum probability (Fig 6.b). The same perfect accordance of the theory and the experimental results are also obtained in the case of random matrices, the elements of which are subjected to the Gaussian distribution with a zero mean.

The work supported by RFBR grant # 06-01-00109.

REFERENCES

Amit, D.J., Gutfreund, H., Sompolinsky, H., 1985. Spin-glass models of neural networks. Physical Review A, v.32, pp.1007-1018.

Fu, Y., Anderson, P.W., 1986. Application of statistical mechanics to NP-complete problems in combinatorial optimization. Journal of Physics A. , v.19, pp.1605-1620.

Hartmann, A.K., Rieger, H., 2004. New Optimization Algorithms in Physics., Wiley-VCH, Berlin.

Hopfield, J.J. 1982. Neural Networks and physical systems with emergent collective computational abilities. Proc. Nat. Acad. Sci.USA. v.79, pp.2554-2558 .

Hopfield, J.J., Tank, D.W., 1985. Neural computation of decisions in optimization problems. Biological Cybernetics, v.52, pp.141-152.

Huajin Tang; Tan, K.C.; Zhang Yi, 2004. A columnar competitive model for solving combinatorial optimization problems. IEEE Trans. Neural Networks v.15, pp.1568 – 1574.

Joya, G., Atencia, M., Sandoval, F., 2002. Hopfield Neural Networks for Optimization: Study of the Different Dynamics. Neurocomputing, v.43, pp. 219-237.

Kryzhanovsky, B., Magomedov, B., 2005. Application of domain neural network to optimization tasks. Proc. of ICANN'2005. Warsaw. LNCS 3697, Part II, pp.397-403.

Kryzhanovsky, B., Magomedov, B., Fonarev, A., 2006. On the Probability of Finding Local Minima in Optimization Problems. Proc. of International Joint Conf. on Neural Networks IJCNN-2006 Vancouver, pp.5882-5887.

Kwok, T., Smith, K.A., 2004. A noisy self-organizing neural network with bifurcation dynamics for combinatorial optimization. IEEE Trans. Neural Networks v.15, pp.84 – 98.

Perez-Vincente, C.J., 1989. Finite capacity of sparce-coding model. Europhys. Lett., v.10, pp.627-631.

Poggio, T., Girosi, F., 1990. Regularization algorithms for learning that are equivalent to multilayer networks. Science 247, pp.978-982.

Salcedo-Sanz, S.; Santiago-Mozos, R.; Bousono-Calzon, C., 2004. A hybrid Hopfield network-simulated annealing approach for frequency assignment in satellite communications systems. IEEE Trans. Systems, Man and Cybernetics, v. 34, 1108 – 1116

Smith, K.A. 1999. Neural Networks for Combinatorial Optimization: A Review of More Than a Decade of Research. INFORMS Journal on Computing v.11 (1), pp.15-34.

Wang, L.P., Li, S., Tian F.Y, Fu, X.J., 2004. A noisy chaotic neural network for solving combinatorial optimization problems: Stochastic chaotic simulated annealing. IEEE Trans. System, Man, Cybern, Part B - Cybernetics v.34, pp. 2119-2125.

Wang, L.P., Shi, H., 2006: A gradual noisy chaotic neural network for solving the broadcast scheduling problem in packet radio networks. IEEE Trans. Neural Networks, vol.17, pp.989 - 1000.

DC MOTOR FAULT DIAGNOSIS BY MEANS OF ARTIFICIAL NEURAL NETWORKS

Krzysztof Patan, Józef Korbicz and Gracjan Głowacki
Institute of Control and Computation Engineering, University of Zielona Góra
ul. Podgórna 50, 65-246 Zielona Góra, Poland
k.patan@issi.uz.zgora.pl, j.korbicz@issi.uz.zgora.pl

Keywords: Neural networks, DC motor, modelling, density shaping, fault detection, fault isolation, fault identification.

Abstract: The paper deals with a model-based fault diagnosis for a DC motor realized using artificial neural networks. The considered process was modelled by using a neural network composed of dynamic neuron models. Decision making about possible faults was performed using statistical analysis of a residual. A neural network was applied to density shaping of a residual, and after that, assuming a significance level, a threshold was calculated. Moreover, to isolate faults a neural classifier was developed. The proposed approach was tested in a DC motor laboratory system at the nominal operating conditions as well as in the case of faults.

1 INTRODUCTION

Electrical motors play a very important role in the safe and efficient work of modern industrial plants and processes. An early diagnosis of abnormal and faulty states renders it possible to perform important preventing actions and it allows one to avoid heavy economic losses involved in stopped production, replacement of elements or parts (Patton et al., 2000). To keep an electrical machine in the best condition, a several techniques like fault monitoring or diagnosis should be implemented. Conventional DC motors are very popular, because they are reasonably cheap and easy to control. Unfortunately, their main drawback is the mechanical collector which has only a limited life spam. In addition, brush sparking can destroy the rotor coil, generate EMC problems and reduce the insulation resistance to an unacceptable limit (Moseler and Isermann, 2000). Moreover, in many cases electrical motors operate in the closed-loop control, and small faults often remain hidden by the control loop. Only if the whole device fails the failure becomes visible. Therefore, there is a need to detect and isolate faults as early as possible.

Recently, a great deal of attention has been paid to electrical motor fault diagnosis (Moseler and Isermann, 2000; Xiang-Qun and Zhang, 2000; Fuessel and Isermann, 2000). In general, elaborated solutions can be splitted into three categories: signal analysis methods, knowledge-based methods and model-based approaches (Xiang-Qun and Zhang, 2000; Korbicz et al., 2004). Methods based on signal analysis include vibration analysis, current analysis, etc. The main advantage of these approaches is that accurate modelling of a motor is avoided. However, these methods only use output signals of a motor, hence the influence of an input on an output is not considered. In turn, the frequency analysis is time-consuming, thus it is not proper for on-line fault diagnosis. In the case of vibration analysis there are serious problems with noise produced by environment and coupling of sensors to the motor (Xiang-Qun and Zhang, 2000).

Knowledge-based approaches are generally based on expert or qualitative reasoning (Zhang and Ellis, 1991). Several knowledge based fault diagnosis approaches have been proposed. These include the rule-based approaches where diagnostic rules can be formulated from process structure and unit functions, and the qualitative simulation-based approaches. The trouble with the use of such models is that accumulating experience and expressing it as knowledge rules is difficult and time-consuming. Therefore, development of a knowledge-based diagnosis system is generally effort demanding.

Model-based approaches include parameter estimation, state estimation, etc. This kind of methods can be effectively used to on-line diagnosis, but its disadvantage is that an accurate model of a motor is required (Korbicz et al., 2004). An alternative solution can be obtained through artificial intelligence, e.g. neural networks. The self-learning ability and property of modelling nonlinear systems allow one to employ neural networks to model complex, unknown and nonlinear dynamic processes (Frank and Köppen-

Seliger, 1997).

The paper is organized as follows. First, description of the considered DC motor is presented in Section 2. The model-based fault diagnosis concept is discussed in Section 3. The architecture details of the dynamic neural network used as a model of the DC motor is given in Section 4. Next, a density shaping technique used to fault detection and a neural classifier applied to isolation of faults are explained in Section 5. Section 6 reports experimental results.

2 AMIRA DR300 LABORATORY SYSTEM

The AMIRA DR300 laboratory system shown in Fig. 1 is used to control the rotational speed of a DC motor with a changing load. The considered laboratory object consists of five main elements: the DC motor M1, the DC motor M2, two digital increamental encoders and the clutch K. The input signal of the engine M1 is an armature current and the output one is the angular velocity. Available sensors for the output are an analog tachometer on optical sensor, which generates impulses that correspond to the rotations of the engine and a digital incremental encoder. The shaft of the motor M1 is connected with the identical motor M2 by the clutch K. The second motor M2 operates in the generator mode and its input signal is an armature current. Available measuremets of the plant are as follows:

- motor current I_m – the motor current of the DC motor M1,
- generator current I_g – the motor current of the DC motor M2,
- tachometer signal T,

and control signals:

- motor control signal C_m – the input of the motor M1,

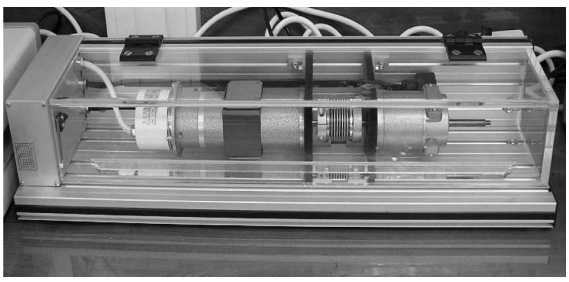


Figure 1: Laboratory system with a DC motor.

Table 1: Laboratory system technical data.

Variable	Value
Motor	
rated voltage	24 V
rated current	2 A
rated torque	0.096 Nm
rated speed	3000 rpm
voltage constant	6.27 mV/rpm
moment of inertia	17.7×10^{-6} Kgm^2
torque constant	0.06 Nm/A
resistance	3.13 Ω
Tachometer	
output voltage	5 mV/rpm
moment of interia	10.6×10^{-6} Kgm^2
Clutch	
moment of inertia	33×10^{-6} Kgm^2
Incremental encoder	
number of lines	1024
max. resolution	4096/R
moment of inertia	1.45×10^{-6} Kgm^2

- generator control signal C_g – the input of the motor M2.

The technical data of the laboratory system is presented in Table 1. The separately excited DC motor is governed by two differential equations. The electrical subsystem can be described by the equation:

$$u(t) = Ri(t) + L\frac{di(t)}{dt} + e(t) \qquad (1)$$

where $u(t)$ is the motor armature voltage, R – the armature coil resistance, $i(t)$ – the motor armature current, L – the motor coil inductance, and $e(t)$ – the induced electromotive force. The induced electromotive force is proportional to the angular velocity of the motor: $e(t) = K_e\omega(t)$, where K_e stands for the motor voltage constant and $\omega(t)$ – the angular velocity of the motor. In turn, the mechanical subsystem can be derived from the torque balance:

$$J\frac{d\omega(t)}{dt} = T_m(t) - B_m\omega(t) - T_l - T_f(\omega(t)) \qquad (2)$$

where J is the motor moment of inertia, T_m – the motor torque, B_m – the viscous friction torque coefficient, T_l – the load torque, and $T_f(\omega(t))$ – the friction torque. The motor torque $T_m(t)$ is proportional to the armature current: $T_m(t) = K_m i(t)$, where K_m stands for the motor torque constant. The friction torque can be considered as a function of angular velocity and it is assumed to be the sum of Stribeck, Coulumb and viscous components. The viscous friction torque opposes motion and it is proportional to the angular velocity. The Coulomb friction torque is constant at any angular velocity. The Stribeck friction is a nonlinear component occuring at low angular velocities.

The model (1)-(2) has a direct relation to the motor physical parameters, however, the relation between them is nonlinear.There are many nonlinear factors in the motor, e.g. the nonlinearity of the magnetization chracteristic of material, the effect of material reaction, the effect caused by eddy current in the magnet, the residual magnetism, the commutator characteristic, mechanical frictions (Xiang-Qun and Zhang, 2000). Summarizing, a DC motor is a nonlinear dynamic process.

3 MODEL-BASED FAULT DIAGNOSIS

Model-based approaches generally utilise results from the field of control theory and are based on parameter estimation or state estimation (Patton et al., 2000; Korbicz et al., 2004). When using this approach, it is essential to have quite accurate models of a process. Generally, fault diagnosis procedure consists of two separate stages: residual generation and residual evaluation. The residual generation process is based on a comparison between the output of the system and the output of the constructed model. As a result, the difference or so-called residual **r** is expected to be near zero under normal operating conditions, but on the occurrence of a fault, a deviation from zero should appear. Unfortunately, designing mathematical models for complex nonlinear systems can be difficult or even impossible. For the model-based approach, the neural network replaces the analytical model that describes the process under normal operating conditions (Frank and Köppen-Seliger, 1997). First, the network has to be trained for this task. Learning data can be collected directly from the process, if possible or from a simulation model that is as realistic as possible. Training process can be carried out off-line or on-line (it depends on availability of data). After finishing the training, a neural network is ready for on-line residual generation. In order to be able to capture dynamic behaviour of the system, a neural network should have dynamic properties, e.g. it should be a recurrent network (Korbicz et al., 2004; Patan and Parisini, 2005). In turn, the residual evaluation block transforms the residual **r** into the fault vector **f** in order to determine the decision about faults, their location and size, and time of fault occurence.

In general, there are three phases in the diagnostic process: detection, isolation and identification of faults (Patton et al., 2000; Korbicz et al., 2004). In practice, however, the identification phase appears rarely and sometimes it is incorporated into fault isolation. Thus, from practical point of view, the diagnostic process consists of two phases only: fault detection and isolation. The main objective of fault detection is to make a decision whether a fault occur or not. The fault isolation should give an information about fault location. Neural networks can be very usefull in designing the residual evaluation block. In the following sections some solutions for realization fault detection and isolation are discussed.

4 DYNAMIC NEURAL NETWORK

Let us consider the neural network described by the following formulae:

$$\varphi_j^1(k) = \sum_{i=1}^{n} w_{ji}^1 u_i(k) \tag{3}$$

$$z_j^1(k) = \sum_{i=0}^{r_1} b_{ji}^1 \varphi_j^1(k-i) - \sum_{i=1}^{r_1} a_{ji}^1 z_j^1(k-i) \tag{4}$$

$$y_j^1(k) = f(z_j^1(k)) \tag{5}$$

$$\varphi_j^2(k) = \sum_{i=1}^{v_1} w_{ji}^2 y_i^1(k) + \sum_{i=1}^{n} w_{ji}^u u_i(k) \tag{6}$$

$$y_j^2(k) = -\sum_{i=1}^{r_2} a_{ji}^2 y_j^1(k-i) \tag{7}$$

$$y_j(k) = \sum_{i=1}^{v_2} w_{ji}^3 y_j^2(k) \tag{8}$$

where w_{ji}^1, $j = 1,\ldots,v_1$ are the input weights, w_{ji}^2, $j = 1,\ldots,v_2$ – the weights between the output and the first hidden layers, w_{ji}^3, $j = 1,\ldots,m$ – the weights between the output and the second hidden layers, w_{ji}^u, $j = 1,\ldots,v_2$ – the weights between output and input layers, $\varphi_j^1(k)$ and $\varphi_j^2(k)$ are the weighted sums of inputs of the j-th neuron in the hidden and output layers, respectively, $u_i(k)$ are the external inputs to the network, $z_j^1(k)$ is the activation of the j-th neuron in the first hidden layer, a_{ji}^1 and a_{ji}^2 are feedback filter parameters of the j-th neuron in the first and second hidden layers, respectively, b_{ji}^1 are the feed-forward filter parameters of the j-th neuron of the first hidden layer, $f(\cdot)$ – nonlinear activation function, $y_j^1(k)$ and $y_j^2(k)$ – the outputs of the j-th neuron of the first and second hidden layers, respectively, and finally $y_j(k)$, $j = 1,\ldots,v_2$ are the network ouputs. The structure of the network is presented in Fig. 2. The first hidden layer consists of v_1 neurons with infinite impulse response filters of the order r_1 and nonlinear activation functions. The second hidden layer consists of v_2 neurons with finite impulse response filters of the order r_2 and linear activation functions. Neurons of this layer receive excitation not only from the neu-

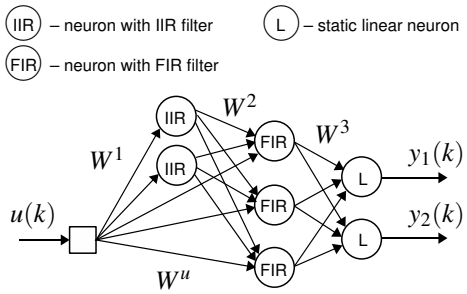


Figure 2: The cascade structure of the dynamic neural network.

rons of the hidden layer but also from the external inputs (Fig. 2). Finaly, the network output is produced by the linear output layer. The network structure (3)-(8) is not a strict feed-forward one. It has a cascade structure. Detailed analysis of this network and its approximation abilities are given in paper (Patan, 2007). Introduction of an additional weight matrix W^u renders it possible to obtain a system, in which the whole state vector is available from the neurons of the output layer. In this way, the proposed neural model can produce a state vector at its output and can serve as a nonlinear observer. This fact is of a crucial importance taking into account training of the neural network. Moreover, this neural structure has similar approximation abilities as a locally recurrent globally feedforward network with two hidden layers designed using nonlinear neuron models with IIR filters, while the number of adaptable parameters is significantly lower (Patan, 2007).

4.1 Network Training

All unknown network parameters can be represented by a vector θ. The objective of training is to find the optimal vector of parameters $\theta^\star$ by minimization of some loss (cost) function:

$$\theta^\star = \arg\min_{\theta \in C} J(\theta) \tag{9}$$

where $J : \mathbb{R}^p \to \mathbb{R}$ represents some loss function to be minimized, p is the dimension of the vector θ, and $C \subseteq \mathbb{R}^p$ is the set of admissible parameters constituted by constraints. To minimize (9) one can use the Adaptive Random Search algorithm (ARS) (Walter and Pronzato, 1996). Assuming that the sequence of solutions $\hat{\theta}_0, \hat{\theta}_1, \ldots, \hat{\theta}_k$ is already appointed, the next point $\hat{\theta}_{k+1}$ is calculated as follows (Walter and Pronzato, 1996):

$$\hat{\theta}_{k+1} = \hat{\theta}_k + \mathbf{r}_k \tag{10}$$

where $\hat{\theta}_k$ is the estimate of the $\theta^\star$ at the k-th iteration, and $\mathbf{r}_k$ is the perturbation vector generated randomly according to the normal distribution $\mathcal{N}(0, \sigma)$. The new point $\hat{\theta}_{k+1}$ is accepted when the cost function $J(\hat{\theta}_{k+1})$ is lower than $J(\hat{\theta}_k)$ otherwise $\hat{\theta}_{k+1} = \hat{\theta}_k$. To start the optimization procedure, it is necessary to determine the initial point $\hat{\theta}_0$ and the variance σ. Let $\theta^\star$ be a global minimum to be located. When $\hat{\theta}_k$ is far from $\theta^\star$, $\mathbf{r}_k$ should have a large variance to allow large displacements, which are necessary to escape the local minima. On the other hand, when $\hat{\theta}_k$ is close $\theta^\star$, $\mathbf{r}_k$ should have a small variance to allow exact exploration of parameter space. The idea of the ARS is to alternate two phases: variance-selection and variance-exploitation (Walter and Pronzato, 1996). During the variance-selection phase, several successive values of σ are tried for a given number of iteration of the basic algorithm. The competing σ_i is rated by their performance in the basic algorithm in terms of cost reduction starting from the same initial point. Each σ_i is computed according to the formula:

$$\sigma_i = 10^{-i}\sigma_0, \quad \text{for} \quad i = 1, \ldots, 4 \tag{11}$$

and it is allowed for $100/i$ iterations to give more trails to larger variances. σ_0 is the initial variance and can be determined, e.g. as a spread of the parameters domain:

$$\sigma_0 = \theta_{max} - \theta_{min} \tag{12}$$

where θ_{max} and θ_{min} are the largest and lowest possible values of parameters, respectively. The best σ_i in terms the lowest value of the cost function is selected for the variance-exploitation phase. The best parameter set $\hat{\theta}_k$ and the variance σ_i are used in the variance-exploitation phase, whilst the algorithm (10) is run typically for one hundred iterations. The algorithm can be terminated when the maximum number of algorithm iteration n_{max} is reached or when the assumed accuracy J_{min} is obtained. Taking into account local minima, the algorithm can be stopped when σ_4 has been selected a given number of times. It means that algorithm got stuck in a local minimum and cannot escape its basin of attraction. Apart from its simplicity, the algorithm possesses the property of global convergence. Moreover, adaptive parameters of the algorithm, cause that a chance to get stuck in local minima is decreased.

5 NEURAL NETWORKS BASED DECISION MAKING

5.1 Fault Detection

In many cases, the residual evaluation is based on the assumption that the underlying data is normally distributed (Walter and Pronzato, 1996). This weak assumption can cause inaccurate decision making and a

number of false alarms can be considerable. Thus, in order to select a threshold in a proper way, the distribution of the residual should be discovered or transformation of the residual to another known distribution should be performed. A transformation of a random vector **x** of an arbitrary distribution to a new random vector **y** of different distribution can be realized by maximazing the output entropy of the neural network (Haykin, 1999; Roth and Baram, 1996; Bell and Sejnowski, 1995). Let us consider a situation when a single input is passed through a transforming function $f(x)$ to give an output y, where $f(x)$ is monotonically increasing continuous function satisfying $\lim_{x\to+\infty} f(x) = 1$ and $\lim_{x\to-\infty} f(x) = 0$. The probability density function of y satisfies

$$p_y(y) = \frac{p_x(x)}{|\partial y/\partial x|} \tag{13}$$

The entropy of the output $h(y)$ is given by

$$h(y) = -E\{\log\ p_y(y)\} \tag{14}$$

where $E\{\cdot\}$ stands for the expected value. Substituting (13) into (14) it gives

$$h(y) = h(x) + E\left\{\log\left|\frac{\partial y}{\partial x}\right|\right\} \tag{15}$$

The first term on the right can be considered to be unaffected by alternations in parameters of $f(x)$. Therefore, to maximize the entropy of y one need to take into account the second term only.

Let us define the divergence between two density functions as follows

$$D(p_x(x), q_x(x)) = E\left\{\log\frac{p_x(x)}{q_x(x)}\right\} \tag{16}$$

finally one obtains

$$h(y) = -D(p_x(x), q_x(x)) \tag{17}$$

where $q_x(x) = |\partial(y)/\partial(x)|$. The divergence between true density of x ($p_x(x)$) and an arbitrary one $q_x(x)$ is minimized when entropy of y is maximized. The input probability density function is then approximated by $|\partial y/\partial x|$. The simple and elegant way to adjust network parameters in order to maximize the entropy of y was given in (Bell and Sejnowski, 1995). They used the on-line version of stochastic gradient ascent rule

$$\Delta w = \frac{\partial h(y)}{\partial w} = \frac{\partial}{\partial w}\left(\log\left|\frac{\partial y}{\partial x}\right|\right) = \left(\frac{\partial y}{\partial x}\right)^{-1}\frac{\partial}{\partial w}\left(\frac{\partial y}{\partial x}\right) \tag{18}$$

Considering the logistic transfer function of the form:

$$y = \frac{1}{1+\exp(-u)}, \quad u = wx + b \tag{19}$$

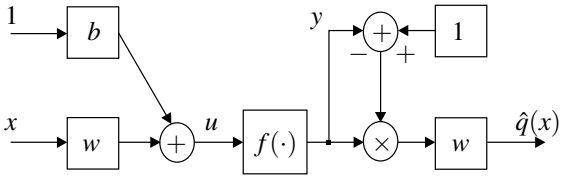


Figure 3: Neural network for density calculation.

where w is the input weight and b is the bias weight, and applying (18) to (19) finally one obtains

$$\Delta w = \frac{1}{w} + x(1+2y) \tag{20}$$

Using a similar reasoning, a rule for the bias weight parameter can be derived

$$\Delta b = 1 - 2y \tag{21}$$

After training, the estimated probability density function can be calculated using scheme shown in Fig. 3. In this case, an output threshold of the density shaping scheme (Fig. 3) can be easily calculated by the formula:

$$T_q = |0.5\alpha(1-0.5\alpha)w| \tag{22}$$

where α is the confidence level.

5.2 Fault Isolation

Transformation of the residual **r** into the fault vector **f** can be seen as a classification problem. For fault isolation, it means that each pattern of the symptom vector **r** is assigned to one of the classes of system behaviour $\{f_0, f_1, f_2, \ldots, f_n\}$. To perform fault isolation, the well-known static multi-layer perceptron network can be used. In fact, the neural classifier should map a relation of the form $\Psi : \mathbb{R}^n \to \mathbb{R}^m : \Psi(\mathbf{r}) = \mathbf{f}$.

5.3 Fault Identification

When analytical equations of residuals are unknown, the fault identification consists in estimating the fault size and time of fault occurence on the basis of residual values. An elementary index of the residual size assigned to the fault size is the ratio of the residual value r_j to suitably assigned threshold value T_j. This threshold can be calculated using (22). In this way, the fault size can be represented as the mean value of such elementary indices for all residuals as follows:

$$s(f_k) = \frac{1}{N}\sum_{j:r_j\in R(f_k)}\frac{r_j}{T_j} \tag{23}$$

where $s(f_k)$ represents the size of the fault f_k, $R(f_k)$ – the set of residuals sensitive to the fault f_k, N – the size of the set $R(f_k)$.

5.4 Evaluation of the Fdi System

The benchmark zone is defined from the benchmark start-up time t_{on} to the benchmark time horizon t_{hor} (Fig. 4). Decisions before the benchmark start-up t_{on} and after the benchmark time horizon t_{hor} are out of interest. The time of fault start-up is represented by t_{from}. When a fault occurs in the system, a residual should deviate from the level assigned to the fault-free case (Fig. 4). The quality of the fault detection system can be evaluated using a number of performance indexes (Patan and Parisini, 2005):

- Time of fault detection t_{dt} – a period of time needed for detection of a fault measured from t_{from} to a permanent, true decision about a fault, as presented in Fig. 4. As one can see there, the first three true decisions are temporary ones and are not taken into account during determining t_{dt}.
- False detection rate r_{fd}:

$$r_{fd} = \frac{\sum_i t^i_{fd}}{t_{from} - t_{on}}, \tag{24}$$

  where t^i_{fd} is the period of i-th false fault detection. This index is used to check the system in the fault-free case. Its value shows a percentage of false alarms. In the ideal case (no false alarms) its value should be equal to 0.
- True detection rate r_{td}:

$$r_{td} = \frac{\sum_i t^i_{td}}{t_{hor} - t_{from}}, \tag{25}$$

  where t^i_{td} is the period of i-th true fault detection. This index is used in the case of faults and describes efficiency of fault detection. In the ideal case (fault detected immediately and surely) its value is equal to 1.

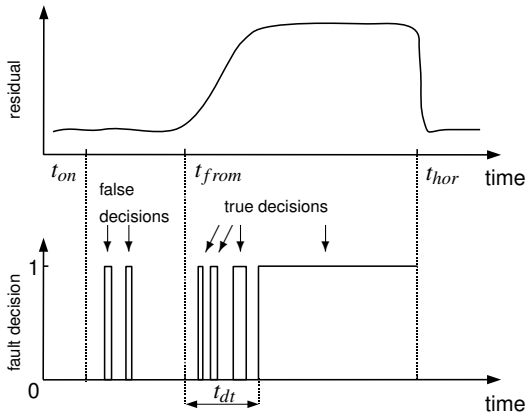


Figure 4: Illustration of performance indexes.

6 EXPERIMENTS

The motor described in Section 2 works is a closed loop control with the PI controller. It is assumed that load of the motor is equal to 0. The objective of the system control is to keep the rotational speed at the constant value equal to 2000. Additionally, it is assumed that the reference value is corrupted by additive noise.

6.1 Motor Modelling

A separately excited DC motor was modelled by using dynamic neural network presented in Section 4. The model of the motor was selected as follows:

$$T = f(C_m) \tag{26}$$

The following input signal was used in experiments:

$$\begin{aligned} C_m(k) = {} & 3\sin(2\pi 0.017k) + 3\sin(2\pi 0.011k - \pi/7) \\ & + 3\sin(2\pi 0.003k + \pi/3) \end{aligned} \tag{27}$$

The input signal (27) is persistantly exciting of the order 6. Using (27) a learning set containig 1000 samples was formed. The neural network model (3)-(8) had the following structure: one input, 3 IIR neurons with the first order filters and hyperbolic tangent activation functions, 6 FIR neurons with the first order filters and linear activation functions, and one linear output neuron. Training process was carried out for 100 steps using the ARS algorithm with initial variance $\sigma_0 = 0.1$. The outputs of the neural model and separately excited motor generated for another 1000 testing samples are depicted in Fig. 5. The efficiency of the neural model was also checked during the work of the motor in the closed-loop control. The results are presented in Fig. 6. After transitional oscilations, the neural model settled at a proper value. This gives a strong argument that neural model mimics the behaviour of the DC motor pretty well.

6.2 Fault Detection

Two types of faults were examined during experiments:

- f_{1_i} – tachometer faults were simulated by increasing/decreasing rotational speed by $\pm 25\%$, $\pm 10\%$ and $\pm 5\%$,
- f_{2_i} – mechanical faults were simulated by increasing/decreasing motor torque by $\pm 40\%$, $\pm 20\%$.

In result, a total of 10 faulty situation were investigated. Using the neural model of the process, a residual signal was generated. This signal was used to

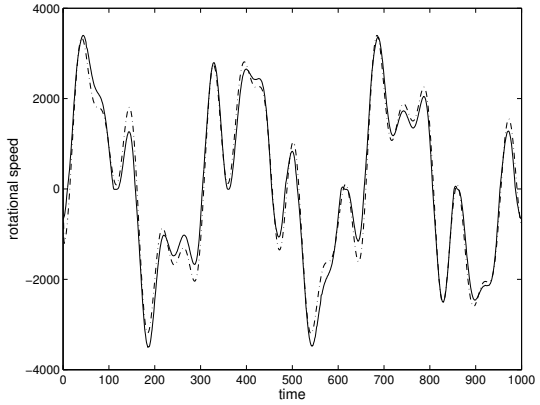


Figure 5: Responses of the motor (solid) and neural model (dash-dot) – open-loop control.

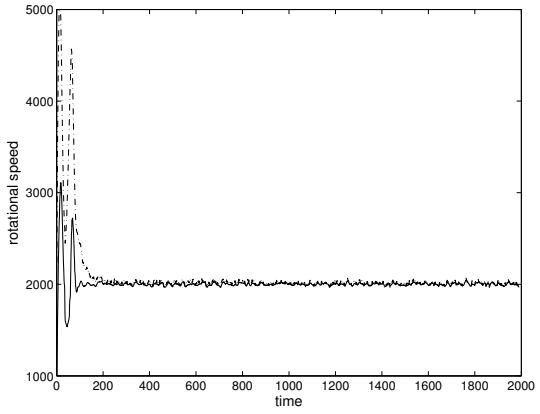


Figure 6: Responses of the motor (solid) and neural model (dash-dot) – closed-loop control.

train another neural network to approximate a probability density function of the residual. Training process was carried out on-line for 100000 steps using unsupervised learning described in Section 5.1. The final network parameters were: $w = -711,43$ and $b = -1.26$. Cut off values determined for significance level $\alpha = 0.05$ had values $x_l = -0.007$ and $x_r = 0,003$ and the threshold was equal to $T = 17,341$. In order to perform decision about faults, and to determine detection time t_{dt}, a time window with the length $n = 50$ was used. If during the following n time steps the residual exceeded the threshold then a fault was signalled. Application of time-window prevents the situation when a temporary true detection will signal a fault (see Fig. 4). The results of fault detection are presented in the second and third columns of Table 2. All faults were reliably detected except fault f_{1_6}. In this case, the changes simulated on tachometer sensor were poorly observed in the residual.

Table 2: Performance indices.

Fault	r_{td}	t_{dt}	R [%]	s
f_{1_1}	0.9995	2035	99	19.51
f_{1_2}	0.9995	2000	82	12.60
f_{1_3}	0.9988	2097	56	5.78
f_{1_4}	0.9395	2086	84	5.45
f_{1_5}	0.9923	2142	62	4.07
f_{1_6}	0.5245	undetected	54	1.48
f_{2_1}	0.9998	2005	100	52.69
f_{2_2}	1.0	2000	100	29,24
f_{2_3}	0.9997	2005	96	27.30
f_{2_4}	0.9993	2008	97	14.56

6.3 Fault Isolation

Fault isolation can be considered as a classification problem where a given residual value is assigned to one of the predefined classes of system behaviour. In the case considered here, there is only one residual signal and 10 different faulty scenarios. To perform fault isolation the well-known multilayer perceptron was used. The neural network had one input (residual signal) and 4 outputs (each class of system behaviour was coded using 4-bit representation). The learning set was formed using 100 samples per each faulty situation and 100 samples representing the fault-free case, then the size of the learning set was equal to 1100. As the well performing neural classifier, the network with 15 hyperbolic tangent neurons in the first hidden layer, 7 hyperbolic tangent neurons in the second hidden layer, and 4 sigmoidal output neurons was selected. The neural classifier was trained for 500 steps using the Levenberg-Marquardt method. Additionally, the real-valued response of the classifier was transformed to the binary one. The simple idea is to calculate a distance between the classifier output and each predefined class of system behaviour. As a result, a binary representation giving the shortest distance is selected as a classifier binary output. This transformation can be represented as follows:

$$j = \arg\min_i ||x - K_i||, \quad i = 1, \ldots, N_K \tag{28}$$

where x is the real-valued output of the classifier, K_i – the binary representation of the i-th class, N_K – the number of predefined classes of system behaviour, and $||\cdot||$ – the Euclidean distance. Then, the binary representation of the classifier can be determined in the form $\bar{x} = K_j$. Recognition accuracy (R) results are presented in the fourth column of Table 2. The worst results 56% and 62% were obtained for the faults f_{1_3} and f_{1_6}, respectively. The fault f_{1_3} was frequently recognized as the fault f_{1_5}. In spite of misrecognizing, this fault was detected as a faulty situation. Quite different situation was observed for the classification of

fault f_{1_6}. This fault, in majority of cases, was classified as the normal operating conditions, thus it cannot be either detected or isolated properly.

6.4 Fault Identification

In this work, the objective of fault identification was to estimate the size of detected and isolated faults. The sizes of faults were calculated using (23). The results are shown in the last column of Table 2. Analyzing results, one can observe that quite large values were obtained for the faults f_{2_1}, f_{2_2} and f_{2_3}. That is the reason that for these three faults true detection rate and recognition accuracy had maximum or close to maximum values. Another group is formed by the faults f_{1_1}, f_{1_2} and f_{2_4} possessing the similar values of the fault size. The third group of fault consists of f_{1_3}, f_{1_4}, and f_{1_5}. The fault sizes in these cases are distinctly smaller than in the cases already discussed. Also the detection time t_{dt} is relatively longer. In spite of the fact that f_{1_6} was not detected, the size of this fault was also calculated. As one can see the fault size in this case is very small, what explains the problems with its detection and isolation.

7 FINAL REMARKS

In the paper, the neural network based method for the fault detection and isolation of faults in a DC motor was proposed. Using the novel structure of dynamic neural network, quite accurate model of the motor was obtained what rendered it possible detection, isolation or even identification of faults. The approach was successfully tested on a number of faulty scenarios simulated in the real plant. The achieved results confirm usefullness and effectiveness of neural networks in designing fault detection and isolation systems. It should be pointed out that presented solution can be easily applied to on-line fault diagnosis.

ACKNOWLEDGEMENTS

This work was supported in part by the Ministry of Science and Higher Education in Poland under the grant *Artificial Neural Networks in Robust Diagnostic Systems*

G. Głowacki is a stypendist of the Integrated Regional Operational Programme (Measure 2.6: Regional innovation strategies and the transfer of knowledge) co-financed from the European Social Fund.

REFERENCES

Bell, A. J. and Sejnowski, T. J. (1995). An information-maximization approach to blind separation and blind deconvolution. *Neural computation*, 7:1129–1159.

Frank, P. M. and Köppen-Seliger, B. (1997). New developments using AI in fault diagnosis. *Artificial Intelligence*, 10(1):3–14.

Fuessel, D. and Isermann, R. (2000). Hierarchical motor diagnosis utilising structural knowledge and a self-learning neuro-fuzzy scheme. *IEEE Trans. Industrial Electronics*, 47:1070–1077.

Haykin, S. (1999). *Neural Networks. A comprehensive foundation, 2nd Edition*. Prentice-Hall, New Jersey.

Korbicz, J., Kościelny, J. M., Kowalczuk, Z., and Cholewa, W., editors (2004). *Fault Diagnosis. Models, Artificial Intelligence, Applications*. Springer-Verlag, Berlin.

Moseler, O. and Isermann, R. (2000). Application of model-based fault detection to a brushless dc motor. *IEEE Trans. Industrial Electronics*, 47:1015–1020.

Patan, K. (2007). Approximation ability of a class of locally recurrent globally feedforward neural networks. In *Proc. European Control Conference, ECC 2007, Kos, Greece*. accepted.

Patan, K. and Parisini, T. (2005). Identification of neural dynamic models for fault detection and isolation: the case of a real sugar evaporation process. *Journal of Process Control*, 15:67–79.

Patton, R. J., Frank, P. M., and Clark, R. (2000). *Issues of Fault Diagnosis for Dynamic Systems*. Springer-Verlag, Berlin.

Roth, Z. and Baram, Y. (1996). Multidimensional density shaping by sigmoids. *IEEE Trans. Neural Networks*, 7(5):1291–1298.

Walter, E. and Pronzato, L. (1996). *Identification of Parametric Models from Experimental Data*. Springer, London.

Xiang-Qun, L. and Zhang, H. Y. (2000). Fault detection and diagnosis of permanent-magnet dc motor based on parameter estimation and neural network. *IEEE Trans. Industrial Electronics*, 47:1021–1030.

Zhang, J., P. D. R. and Ellis, J. E. (1991). A self-learning fault diagnosis system. *Transactions of the Institute of Measurements and Control*, 13:29–35.

SATURATION FAULT-TOLERANT CONTROL FOR LINEAR PARAMETER VARYING SYSTEMS

Ali Abdullah
Kuwait University, Electrical Engineering Department, P. O. Box 5969, Safat-13060, Kuwait
alkandary@eng.kuniv.edu.kw

Keywords: Fault diagnosis, fault-tolerant systems, parameter estimation.

Abstract: This paper presents a methodology for designing a fault-tolerant control (FTC) system for linear parameter varying (LPV) systems subject to actuator saturation fault. The FTC system is designed using linear matrix inequality (LMI) and model estimation techniques. The FTC system consists of a nominal control, fault diagnostic, and fault accommodation schemes. These schemes are designed to achieve stability and tracking requirements, estimate a fault, and reduce the fault effect on the system. Simulation studies are used to illustrate the proposed design.

1 INTRODUCTION

In recent years, the field of designing FTC systems has received considerable attention (Blanke et al., 2001; Bodson, 1995; Isermann et al., 2002; Patton, 1997; Rauch, 1994; Stengel, 1991). For the case of actuator fault, most of this research had addressed fault accommodation for system subject to parameter variation or frozen output. Other types of actuator fault have been rarely considered. In this paper, a methodology for designing FTC system for LPV systems subject to a reduction in the actuator saturation limit is presented. The LPV systems are defined as a class of linear time-varying systems whose state space matrices depend on a set of parameters that are bounded and can be measured or estimated online.
In the case of using an analytical approach, the main idea behind fault tolerance is the use of fault diagnostic and accommodation schemes. A fault diagnostic scheme driven by plant measurements is used to detect, locate, and estimate faults; while a fault accommodation scheme driven by fault information from the diagnostic scheme is used to modify the nominal control law in order to reduce the fault effect on the system. Based on the above idea, the total task of the proposed FTC system is divided into three parts:

- Plant control: attempts to stabilize the closed-loop system and provide the desired tracking properties in the absence of faults. The controller is designed using LPV technique (Apkarian et al., 1995; Apkarian and Adams, 1998; Gahinet et al., 1996; Kose et al., 1998; Tuan and Apkarian, 2002).

- Fault diagnosis: deals with the problem of saturation fault detection, location, and level estimation. To achieve that, a suitable LPV model is derived to describe the faulty system. Then the results in (Polycarpou and Helmicki, 1995) are used to construct the diagnostic scheme.

- Fault accommodation: attempts to reduce the fault effect on the system by modifying the nominal control law through the reference reshaping filter and feed-forward gain. The accommodation scheme is designed with the help of the bounded real lemma for LPV system presented in (Gahinet et al., 1996).

The notation $\mathcal{H}(\mathcal{A},\mathcal{B},\mathcal{C},\mathcal{D},\mathcal{E},\mathcal{F})$ is used throughout the paper to denote the symmetric matrix $\begin{pmatrix} \mathcal{A} & \mathcal{B} & \mathcal{C} \\ \mathcal{B}^T & \mathcal{D} & \mathcal{E} \\ \mathcal{C}^T & \mathcal{E}^T & \mathcal{F} \end{pmatrix}$.

2 PROBLEM STATEMENT

Consider a class of LPV systems of the form:

$$\dot{x}_s(t) = A_s(\rho(t))x_s(t) + B_s u(t) \quad (1)$$
$$y(t) = x_s(t) \quad (2)$$

where $x_s(t) \in \Re^n$ is the state vector, $u(t) \in \Re^m$ is the control signal, and $y(t) \in \Re^n$ is the measured output signal. $A_s(\rho(t)) = A_{s_o} + \sum_{i=1}^{N} \rho_i(t) A_{s_i}$, and A_{s_j} and B_s are known constant matrices with appropriate dimensions. Furthermore, $\rho(t) = (\rho_1(t), \ldots, \rho_N(t))^T$ is the vector of real time varying parameters ranging inside the hyper-rectangle region defined by $\rho_i(t) \in [\underline{\rho}_i, \overline{\rho}_i]$. Also, its rate $\dot{\rho}(t) = (\dot{\rho}_1(t), \ldots, \dot{\rho}_N(t))^T$ is ranging inside another hyper-rectangle region defined by $\dot{\rho}_i(t) \in [\underline{\nu}_i, \overline{\nu}_i]$.

The actuator saturation fault considered in this study is given by Definition 1.

Definition 1 (Actuator Saturation Fault) The actuator saturation fault is defined mathematically as:

$$\sigma_j(\delta_j u_j) = \begin{cases} u_j & |u_j| < \delta_j \overline{u}_j \\ \delta_j \overline{u}_j \, sign(u_j) & |u_j| \geq \delta_j \overline{u}_j \end{cases}$$

where u_j is the input to the jth actuator, σ_j is the output of the jth actuator, and $0 \leq \delta_j \leq 1$ is the reduced level of the jth saturation limit $\overline{u}_j$.

Remark 1 The value of δ_j represents the reduced level of the actuator saturation limit where $\delta_j = 0$ means a complete failure, $\delta_j = 1$ means no failure exists, and $0 < \delta_j < 1$ means the saturation limit has been reduced to the value of $\pm\delta_j \overline{u}_j$.

Now the main problem is presented.

Problem 1 Design an FTC system for the LPV system (1)-(2) such that:

- In the absence of actuator saturation fault, the nominal control objectives are achieved.
- In the presence of actuator saturation fault, the control objectives are achieved as close as possible to the nominal one.

3 FAULT-TOLERANT CONTROL SYSTEM

The main schemes of the FTC system are shown in Figure 1. These schemes are controller, fault diagnosis, and fault accommodation. Furthermore, the fault accommodation scheme consists of reconfiguration mechanism, feed-forward gain, and reference reshaping filter. The controller is designed to achieve

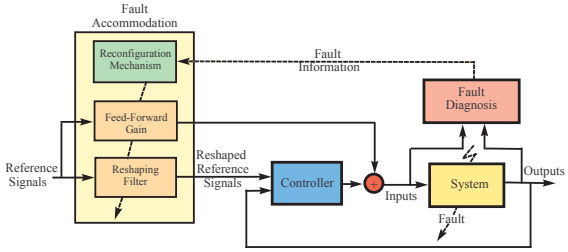


Figure 1: Structure of fault-tolerant control system.

the desired system performances assuming that the system is under normal operation. The fault diagnostic scheme is designed to detect, locate, and estimate a fault. The fault diagnostic scheme is driven by the available system input and output signals. The fault information (no fault, fault, location and magnitude of fault) is supplied to the reconfiguration mechanism to trigger an appropriate reconfiguration of the feed-forward gain and reference reshaping filter. The feed-forward gain and reference reshaping filter are designed to fulfill the new physical constraints imposed by the fault.

3.1 Control Design

The controller is designed based on the concept of affine quadratic stability (Gahinet et al., 1996) defined below.

Definition 2 (Affine Quadratic Stability) The LPV system $\dot{x} = A_c(\rho)x$ is affinely quadratically stable (AQS) if there exists N+1 symmetric matrices P_i such that the following inequalities:

$$P(\rho) = P_o + \rho_1 P_1 + \ldots + \rho_i P_i + \ldots + \rho_N P_N > 0 \quad (3)$$
$$F(\rho, \dot{\rho}) = A_c(\rho)^T P(\rho) + P(\rho) A_c(\rho) + \frac{dP(\rho)}{dt} < 0, \quad (4)$$

where $\frac{dP(\rho)}{dt} = \dot{\rho}_1 P_1 + \ldots + \dot{\rho}_i P_i + \ldots + \dot{\rho}_N P_N$, hold for all admissible trajectories of the parameter vector ρ. In this case, the function $V(x, \rho) = x^T P(\rho) x$ is a quadratic Lyapunov function for the LPV system $\dot{x} = A_c(\rho)x$.

The difficulty associated with the control design using Definition 2 is that the matrix inequality (4) is not linear in terms of $P(\rho)$ and $A_c(\rho)$. However, Lemma 1 (Bara et al., 2001) can be used instead of Definition 2 to simplify a controller design.

Lemma 1 The LPV system $\dot{x}(t) = A_c(\rho)x(t)$ is AQS if there exist a constant matrix W and a symmetric matrix $P(\rho)$ such that the following LMI:

$$\mathcal{H}\left(-W - W^T, W^T A_c(\rho)^T + P(\rho), W^T,\right.$$

$$-P(\rho)+\frac{dP(\rho)}{dt},0,-P(\rho))<0 \quad (5)$$

holds for all admissible trajectories of the parameter vector ρ.

Since the parameter vector ρ ranges over a polytope, the LMI (5) involves infinite number of constraints. Theorem 1 is used to reduce the infinite number of constraints into a finite one, hence simplifying the controller design.

Theorem 1 If there exist matrices W, R_i, and symmetric matrices P_i such that the following LMIs:

$$\mathcal{H}\,(-W-W^T,(A_s(Ve_i)W-B_sR_i+P(Ve_i))^T,W^T,$$

$$-P(Ve_i)-P_o+P(\tilde{Ve}_i),0,-P(Ve_i))<0 \quad (6)$$

are feasible for all $(Ve_i,\tilde{Ve}_i)$ where $\rho=\sum_{i=1}^{2^N}\alpha_i(t)Ve_i$, and $\dot{\rho}=\sum_{i=1}^{2^N}\beta_i(t)\tilde{Ve}_i$ with $\sum_{i=1}^{2^N}\alpha_i(t)=1$, $\sum_{i=1}^{2^N}\beta_i(t)=1$, $\alpha_i(t)\geq 0$, and $\beta_i(t)\geq 0$. Then, the control law $u=-(\sum_{i=1}^{2^N}\alpha_i(t)K_i)x(t)$ where $K_i=R_iW^{-1}$ stabilizes the system (1)-(2).

Proof: is omitted.

To implement the control law $u=-(\sum_{i=1}^{2^N}\alpha_i(t)K_i)x(t)$, $\alpha_i(t)$ must be available online. $\alpha_i(t)$ can be computed from the relation $\rho=\sum_{i=1}^{2^N}\alpha_i(t)Ve_i$ using the known Ve_i and $\rho(t)$.

3.2 Fault Diagnosis

To diagnosis a fault, consider the case where the saturation limit of the *jth* actuator is reduced due to a fault. Then, equation (1) is written as:

$$\dot{x_s}=A_s(\rho)x_s+B_{s_1}u_1\ldots+B_{s_j}\sigma_j(\delta_ju_j)\ldots+B_{s_m}u_m \quad (7)$$

For fault diagnosis, equation (7) is expressed in terms of control outputs by defining:

$$\lambda_j=\begin{cases}\frac{\sigma_j(\delta_ju_j)}{u_j}-1 & u_j\neq 0\\ 0 & u_j=0\end{cases}$$

and then writing (7) as:

$$\dot{x_s}=\begin{cases}A_s(\rho)x_s(t)+B_su+\lambda_jB_{s_j}u_j & u_j\neq 0\\ A_s(\rho)x_s+B_su & u_j=0\end{cases}$$

Theorem 2 is used to estimate the value of λ_j which will be used to detect the fault and to estimate the saturation level δ_j.

Theorem 2 Consider the estimated model:

$$\dot{\hat{x}}_s=A_s(\rho)x_s+B_su+\hat{\lambda}_jB_{s_j}u_j+G(\hat{x}_s-x_s)$$

where $\hat{x}_s\in\mathbb{R}^n$ is the estimated state vector, G is the constant matrix with negative eigenvalues, and $\hat{\lambda}_j\in\mathbb{R}$ is the estimated parameter of λ_j adjusted as:

$$\dot{\hat{\lambda}}_j=\Gamma(B_{s_j}u_j)^Te-\chi\Gamma\frac{\hat{\lambda}_j\hat{\lambda}_j^T}{|\hat{\lambda}_j|^2}\Gamma(B_{s_j}u_j)^Te;\ \hat{\lambda}_j(0)=0 \quad (8)$$

where $e=x-\hat{x}$ is the estimated error, Γ is the positive definite matrix, and χ is the indicator function for the projection algorithm (to prevent parameter drift) defined as:

$$\chi=\begin{cases}0 & (|\hat{\lambda}_j|<M)\ or\ (|\hat{\lambda}_j|=M\ and\ \hat{\lambda}_j^T\Gamma(B_{s_j}u_j)^Te\leq 0)\\ 1 & (|\hat{\lambda}_j|=M\ and\ \hat{\lambda}_j^T\Gamma(B_{s_j}u_j)^Te>0)\end{cases}$$

Then $\hat{\lambda}_j$ is uniformly bounded, and $lim_{t\longrightarrow\infty}e(t)=0$.

Proof: is omitted.

For fault detection, $\hat{\lambda}_j$ is tested for the likelihood of saturation fault. A decision about the existence of saturation fault is made as follows: if $\nu_j\leq\varepsilon_j$, the saturation fault doest not exist; if $\nu_j>\varepsilon_j$, the saturation fault exist. $\nu_j=[\frac{1}{\alpha}\int_t^{t+\alpha}(\hat{\lambda}_j(\tau))^2d\tau]^{1/2}$ is the average energy of $\hat{\lambda}_j$ over the time interval $[t,t+\alpha]$, α is the detection window, and ε_j is the threshold.
For saturation level estimation, when the saturation fault exists (i.e., $\nu_j>\varepsilon_j$), $\hat{\lambda}_j$ has the value: $\hat{\lambda}_j=\frac{\sigma_j(\hat{\delta}_ju_j)}{u_j}-1=\frac{\hat{\delta}_j\bar{u}_j\,sign(u_j)}{u_j}-1=\frac{\hat{\delta}_j\bar{u}_j}{|u_j|}-1$. Then the saturation level δ_j can be estimated as: $\hat{\delta}_j=\frac{|u_j|}{\bar{u}_j}(\hat{\lambda}_j+1)$.

3.3 Fault Accommodation

To accommodate a saturation fault, a reference reshaping filter and feed-forward gain are used to fulfil the new input constraint $|u_j(t)|<\delta_j\bar{u}_j$. To enforce this constraint the *jth* system input $u_j(t)$ is generated from the *jth* control output $u_{c_j}(t)$, which is a function of a modified reference signal $\bar{r}$ generated by the reference reshaping filter, and the *jth* feed-forward signal u_{f_j} in order to get $|u_j(t)=u_{c_j}(t)+u_{f_j}(t)|<\delta_j\bar{u}_j$. Furthermore, the modified reference signal $\bar{r}$ should be designed to fulfil the input constraint while deviation from the reference signal r is minimized. Based on these suggestions, Problem 2 is addressed as follows.

Problem 2 Given the:

- System: $\begin{cases}\dot{x}_s(t)=A_s(\rho)x_s(t)+B_su(t)\\ y(t)=x_s(t)\end{cases}$
- Controller: $\begin{cases}\dot{x}_c(t)=A_c(\rho)x_c(t)+B_c(\rho)\bar{r}(t)+C_c(\rho)x_s(t)\\ u_c(t)=D_c(\rho)x_c(t)+E_c(\rho)\bar{r}(t)+F_c(\rho)x_s(t)\end{cases}$

- Saturation fault level: δ_j

Design a:

- Reference reshaping filter: $\begin{cases} \dot{x}_r(t) = A_r(\delta_j)x_r(t) + B_r(\delta_j)r(t) \\ \bar{r}(t) = C_r(\delta_j)x_r(t) + D_r(\delta_j)r(t) \end{cases}$
- Feed-forward: $u_f(t) = F(\delta_j)r(t)$

Such that the whole system is stable, $|u_j(t) = u_{c_j}(t) + u_{f_j}(t)| < \delta_j \bar{u}_j$, and $\|\bar{r}(t) - r(t)\|_2$ is minimized.

The reference reshaping filter and feed-forward gain are designed based on the concept of affine quadratic H_∞ performance (Gahinet et al., 1996) defined below.

Definition 3 (Affine Quadratic H_∞ Performance) The LPV system:

$$\dot{x}(t) = A(\rho)x(t) + B(\rho)r(t) \quad (9)$$
$$u_j(t) = C_j(\rho)x(t) + D_j(\rho)r(t) \quad (10)$$

has affine quadratic H_∞ performance γ_j if there exist $N+1$ symmetric matrices P_i such that:

$$P(\rho) = P_o + \rho_1 P_1 + \ldots + \rho_i P_i + \ldots + \rho_N P_N > 0 \quad (11)$$
$$\mathcal{H}\,(A(\rho)P(\rho) + P(\rho)A^T(\rho) + P(\dot{\rho}) - P_o, P(\rho)C_j^T(\rho), B(\rho), -\gamma_j I, D_j(\rho), -\gamma_j I) < 0 \quad (12)$$

holds for all admissible parameter vector ρ. In such a case, the Lyapunov function $V(x,\rho) = x^T P(\rho) x$ establishes that the system (9) is asymptotically stable and its L_2 gain does not exceed γ_j. That is, $|u_j(t)| < \gamma_j \|r(t)\|_2$ for all L_2-bounded $r(t)$ (provided that $x(0) = 0$).

The difficulty of using Definition 3 to design the reference reshaping filter and feed-forward gain resides in the fact that matrix inequality (12) is not linear in terms of $P(\rho)$ and $A(\rho)$. Therefore, the matrix inequality (12) is not convex and thus difficult to solve. To convert the problem into an LMI problem and make it tractable, the following relaxations and selections are proposed:

- To get a LMI, $A_r(\delta_j)$ and $C_r(\delta_j)$ are predesigned and denoted as: $A_r(\delta_j) = A_{r_j}$ and $C_r(\delta_j) = C_{r_j}$, where A_{r_j} has negative eigenvalues
- To satisfy $|u_j(t)| < \delta_j \bar{u}_j$, γ_j should satisfy: $\gamma_j \leq \frac{\delta_j \bar{u}_j}{max(\|r(t)\|_2)}$
- Minimizing $\|\bar{r}(t) - r(t)\|_2$ is relaxed and making $\bar{r}(t) = \Lambda_j r(t)$ is considered at the steady state, that is $D_r(\delta_j) = \Lambda_j + C_{r_j} A_{r_j}^{-1} B_r(\delta_j)$, where Λ_j is a constant diagonal matrix with its elements $0 < \mu_{i,j} \leq 1$
- The structures of the remaining design matrices are selected as: $B_r(\delta_j) = \delta_j B_{r_j}$ and $F(\delta_j) = \delta_j F_j$, where B_{r_j} and F_j are constant design matrices.

The state space representation of the plant with controller, reference reshaping filter, and feed-forward gain is given by:

$$\dot{x}(t) = A(\rho)x(t) + B(\rho)H_j r(t) \quad (13)$$
$$u_j(t) = C_j(\rho)x(t) + D_j(\rho)H_j r(t) \quad (14)$$

where

$x(t) = (x_s^T(t), x_c^T(t), x_r^T(t))^T$, $H_j = (\Lambda_j^T, B_{r_j}^T, F_j^T)^T$,

$$A(\rho) = \begin{pmatrix} A_s(\rho) + B_s F_c(\rho) & B_s D_c(\rho) & B_s E_c(\rho) C_{r_j} \\ C_c(\rho) & A_c(\rho) & B_c(\rho) C_{r_j} \\ 0 & 0 & A_{r_j} \end{pmatrix},$$

$B(\rho) = (((B_s E_c(\rho))^T, B_c^T(\rho), 0)^T \vdots ((\delta_j B_s E_c(\rho) C_{r_j} A_{r_j}^{-1})^T,$

$(\delta_j B_c(\rho) C_{r_j} A_{r_j}^{-1})^T, \delta_j I)^T \vdots (\delta_j B_s^T, 0, 0)^T)$,

$C_j(\rho)$=jth row of $\{(F_c(\rho), D_c(\rho), E_c(\rho) C_{r_j})\}$, and

$D_j(\rho)$=jth row of $\{(E_c(\rho) \vdots (\delta_j E_c(\rho) C_{r_j} A_{r_j}^{-1}) \vdots \delta_j I)\}$.

Corollary 1 is used to convert Problem 2 into an LMI problem.

Corollary 1 Consider the LPV system (13)-(14) with known δ_j, A_{r_j}, and C_{r_j}. If there exists a solution (P_i, H_j, γ_j) that maximize $\mu_{i,j}$ subject to:

$$P(\rho) = P_o + \rho_1 P_1 + \ldots + \rho_i P_i + \ldots + \rho_N P_N > 0 \quad (15)$$
$$\mathcal{H}\,(A(\rho)P(\rho) + P(\rho)A^T(\rho) + P(\dot{\rho}) - P_o, P(\rho)C_j^T(\rho), B(\rho)H_j, -\gamma_j I, D_j(\rho)H_j, -\gamma_j I) < 0 \quad (16)$$
$$\gamma_j \leq \frac{\delta_j \bar{u}_j}{max(\|r(t)\|_2)} \quad (17)$$
$$0 < \mu_{i,j} \leq 1 \quad (18)$$

for all admissible parameter vectors ρ. Then, the system (13) is asymptotically stable, $|u_j(t)| < \delta_j \bar{u}_j$, and $\bar{r}(t) = \Lambda_j r(t)$ as $t \longrightarrow \infty$ (provided that $r(t)$ is a bounded constant reference).

Proof: is omitted.

The LMIs (15)-(16) need to be solved for all admissible parameter vectors ρ which imply infinite number of LMIs. However, the infinite number of LMIs can be reduced to a finite number of LMIs using the following procedures:

- Write the matrices in (13)-(14) as: $A(\rho) = A_o + \sum_{i=1}^N \rho_i A_i$, $B(\rho) = B_o + \sum_{i=1}^N \rho_i B_i$, $C_j(\rho) = C_{j_o} + \sum_{i=1}^N \rho_i C_{j_i}$, and $D_j(\rho) = D_{j_o} + \sum_{i=1}^N \rho_i D_{j_i}$
- Use the matrix expressions in the previous step to write the LMI (16) as:

$$M_o + \sum_{i=1}^N \dot{\rho}_i M_{dot} + \sum_{i=1}^N \rho_i M_i + \sum_{i,l=1, i\neq l}^N \rho_i \rho_l M_{il}$$

$$+\sum_{i=1}^{N}\rho_i^2 M_{ii} < 0 \qquad (19)$$

where $M_o = \mathcal{H}(A_oP_o + P_oA_o^T, P_oC_{j_o}^T, B_oH_j, -\gamma_j I, D_{j_o}H_j, -\gamma_j I)$, $M_{dot} = \mathcal{H}(P_i, 0, 0, 0, 0, 0)$, $M_i = \mathcal{H}(A_oP_i + A_iP_o + P_oA_i^T + P_iA_o^T, P_oC_{j_i}^T + P_iC_{j_o}^T, B_iH_j, 0, D_{j_i}H_j, 0)$, $M_{il} = \mathcal{H}(A_iP_l + A_lP_i + P_iA_l^T + P_lA_i^T, P_iC_{j_l}^T + P_lC_{j_i}^T, 0, 0, 0, 0)$, and $M_{ii} = \mathcal{H}(A_iP_i + P_iA_i^T, P_iC_{j_i}^T, 0, 0, 0, 0)$.

- To reduce the number of parameters ($\dot{\rho}_i$, ρ_i, $\rho_i\rho_l$, and ρ_i^2) and hence to reduce the design complexity, define fewer parameters σ_i, $i = 1, 2, \ldots, K$, to bound the parameters $\dot{\rho}_i$, ρ_i, $\rho_i\rho_l$, and ρ_i^2. Then, the LMIs (15) and (19) will be a function of σ_i.
- Solve the LMIs (15), (17), (18), and (19) for P_i, H_j, and γ_j at all vertices of the σ parameter space. In this case, the existing solutions guarantee the feasibility of the LMIs for all admissible parameter vector σ, see for example (Gahinet et al., 1996).

The following procedures are implemented inside the reconfiguration mechanism scheme in order to adapt the right reference reshaping filter and feed-forward gain after estimating the level of saturation fault.

- If there is no saturation fault, i.e. $v_j(k) \le \varepsilon_j$, set $F_j = 0$ and $\bar{r}(t) = r(t)$, then stop. Otherwise, go to the next step.
- Given the estimated level $\hat{\delta}_j \in S_j$, where $S_j = \{\underline{\kappa}_j \le \hat{\delta}_j \le \overline{\kappa}_j : 0 \le \underline{\kappa}_j, \overline{\kappa}_j \le 1\}$, adapt the reference reshaping filter and feed-forward gain designed using the above procedures $\forall \hat{\delta}_j \in S_j$, then stop.

4 ILLUSTRATION EXAMPLE

In this section a second-order LPV model is used to illustrate the proposed design. The LPV model is given by:

$$\underbrace{\begin{pmatrix}\dot{x}_{s1}\\ \dot{x}_{s2}\end{pmatrix}}_{\dot{x}_s} = \underbrace{\begin{pmatrix}\rho(t) & 1\\ 0 & -1\end{pmatrix}}_{A_s(\rho)}\begin{pmatrix}x_{s1}\\ x_{s2}\end{pmatrix} + \underbrace{\begin{pmatrix}1\\ 0\end{pmatrix}}_{B_s}u \qquad (20)$$

$$y = x_{s1} \qquad (21)$$

where -0.5= $\underline{\rho}$ $\le \rho(t) \le$ $\overline{\rho}$ =0.1.

4.1 Control System

To design a controller using the result of Theorem 1, the LPV model (20) is written in the polytopic form $\dot{x}_s = (\alpha_1 A_s(\overline{\rho}) + \alpha_2 A_s(\underline{\rho}))x_s + B_s u$, where $\alpha_1 = (\rho(t) - \underline{\rho})/(\overline{\rho} - \underline{\rho})$, and $\alpha_2 = (-\rho(t) + \overline{\rho})/(\overline{\rho} - \underline{\rho})$. Then the results of Theorem 1 are used to design the controller to stabilize the closed loop system, and to track the desired constant reference signal. Figure 2 shows that the controlled system output reaches the desired set point with "almost" perfect tracking.

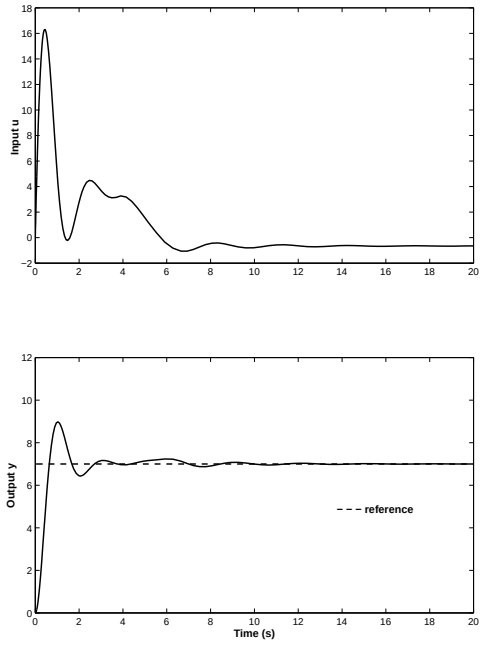


Figure 2: Simulation results of nominal control system.

4.2 Fault Diagnosis

The proposed design of saturation fault diagnosis is simulated for the LPV model (20). Figure 3 shows the results of fault diagnostic system with the saturation level of $\delta = 0.5$. The fault is occurred at 5.00 sec and detected at 6.12 sec using the threshold value of 0.05. The results of Theorem 2 are used to estimate the saturation level. The learning rate Γ is chosen as $\Gamma = 10$, and the filter pole is set to $p = -60$. Figure 3 shows that after the occurrence of the fault, the level of the saturation fault is converged to the true value within about 3 sec. The result indicates that the estimation scheme provides an accurate estimation of the saturation level within a reasonable time.

4.3 Fault Accommodation

Once the saturation level is estimated and sent to the reconfiguration mechanism scheme. An appropriate reconfiguration of the feed-forward gain and refer-

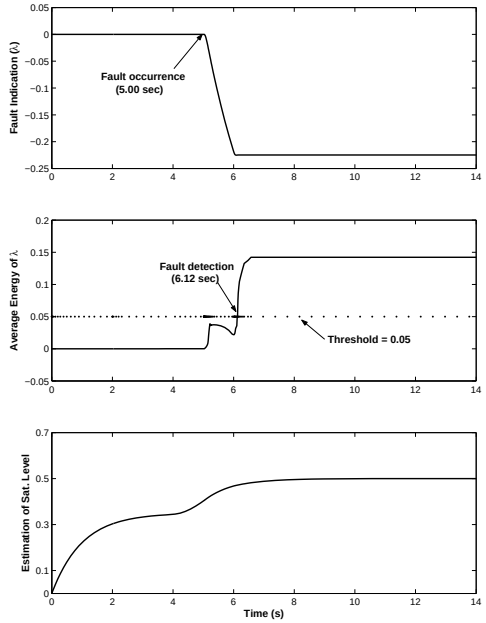


Figure 3: Simulation results of fault diagnostic scheme.

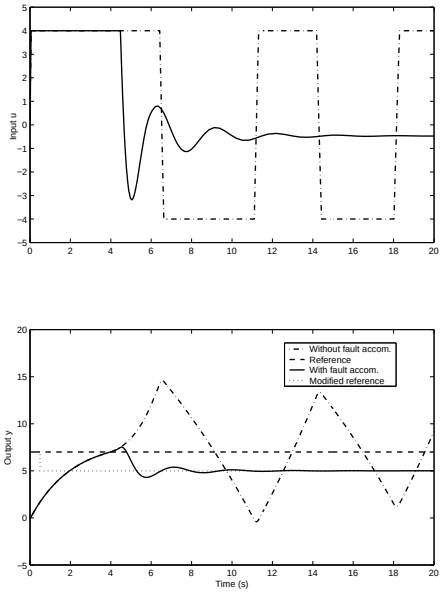


Figure 4: Simulation results with and without fault accommodation.

ence reshaping filter is triggered to accommodate the fault. Figure 4 shows the simulation results of the control system with and without fault accommodation for the case of 0.2 reduction in the saturation limit. Without fault accommodation, the input signal reaches its limit and the output diverges from the desired set point. With fault accommodation, on the other hand, the input signal is reduced within its new limit and the output tracks the desired set point.

5 CONCLUSION

Designing a FTC system for LPV systems subject to actuator saturation fault is considered. The FTC system consists of a nominal control, fault diagnostic, and fault accommodation schemes in order to achieve control objectives in the absence and presence of actuator saturation fault. Simulation results demonstrate the effectiveness of the proposed FTC system.

REFERENCES

Apkarian, P. and Adams, R. J. (1998). Advanced gain-scheduling techniques for uncertain systems. *IEEE Transaction on Control Systems Technology*, 6:21–32.

Apkarian, P., Gahinet, P., and Becker, G. (1995). Self-scheduled h_∞ control of linear parameter-varying systems: a design example. *Automatica*, 31:1251–1261.

Bara, G. I., Daafouz, J., Kratz, F., and Ragot, J. (2001). Parameter-dependent state observer design for affine lpv systems. *International Journal of Control*, 74:1601–1611.

Blanke, M., Staroswiecki, M., and Wu, N. E. (2001). Concepts and methods in fault-tolerant control. In *proc. American Control Conference*.

Bodson, M. (1995). Emerging technologies in control engineering. *IEEE Control Systems Magazine*, 15.

Gahinet, P., Apkarian, P., and Chilali, M. (1996). Affine parameter-dependent lyapunov functions and real parametric uncertainty. *IEEE Transaction on Automatic Control*, 41:436–442.

Isermann, R., Schwarz, R., and Stolzl, S. (2002). Fault-tolerant drive-by-wire systems. *IEEE Control Systems Magazine*, 22.

Kose, I. E., Jabbari, F., and Schmitendorf, W. E. (1998). A direct characterization of l2-gain controllers for lpv systems. *IEEE Transaction on Automatic Control*, 43:1302–1307.

Patton, R. J. (1997). Fault-tolerant control systems: the 1997 situation. In *SAFEROCESS'97, IFAC Symp. Fault Detection, Supervision and Safety for Technical Processes*.

Polycarpou, M. M. and Helmicki, A. J. (1995). Automated fault detection and accommodation: a learning systems approach. *IEEE Transactions on Systems, Man and Cybernetics*, 25.

Rauch, H. E. (1994). Intelligent fault diagnosis and control reconfiguration. *IEEE Control Systems Magazine*, 14.

Stengel, R. F. (1991). Intelligent failure-tolerant control systems. *IEEE Control Systems Magazine*, 11.

Tuan, H. D. and Apkarian, P. (2002). Monotonic relaxations for robust control: new characterizations.

A FUZZY PARAMETRIC APPROACH FOR THE MODEL-BASED DIAGNOSIS

F. Lafont[1], N. Pessel[1] and J. F. Balmat[2]
LSIS, UMR CNRS 6168, University of South-Toulon-Var
[1] IUT of Toulon, [2] Faculty of Sciences and Techniques
B.P 20132, 83957 La Garde Cedex, France
lafont@univ-tln.fr, nathalie.pessel@univ-tln.fr, balmat@univ-tln.fr

Keywords: Adaptive model, fuzzy system models, diagnosis, Fault Detection and Isolation (FDI).

Abstract: This paper presents a new approach for the model-based diagnosis. The model is based on an adaptation with a variable forgetting factor. The variation of this factor is managed thanks to fuzzy logic. Thus, we propose a design method of a diagnosis system for the sensors defaults. In this study, the adaptive model is developed theoretically for the Multiple-Input Multiple-Output (MIMO) systems. We present the design stages of the fuzzy adaptive model and we give details of the Fault Detection and Isolation (FDI) principle. This approach is validated with a benchmark: a hydraulic process with three tanks. Different defaults (sensors) are simulated with the fuzzy adaptive model and the fuzzy approach for the diagnosis is compared with the residues method. The first results obtained are promising and seem applicable to a set of MIMO systems.

1 INTRODUCTION

The automatic control of technical systems requires a fault detection to improve reliability, safety and economy. The diagnosis is the detection, the isolation and the identification of the type as well as the probable cause of a failure using a logical reasoning based on a set of information coming from an inspection, a control or a test (AFNOR, CEI) (Noura, 2002 - Szederkényi, 1998). The model-based diagnosis is largely studied in the literature (Ripoll, 1999 – Maquin, 1997 – Isermann, 1997). These methods are based on parameter estimation, parity equations or state observers. (Ripoll, 1999 - Maquin, 1997 – Isermann, 2005). The goal is to generate the indicators of defaults through the generation of residues (Isermann, 1984).

This paper deals with the problem of the model-based diagnosis by using a parametric estimation method. We particularly focus our study on an approach with an adaptive model. Many methods exist which enable the design of these adaptive models (Ripoll, 1999).

Many works tackle the model-based diagnosis from a fuzzy model of the processes (Querelle et al., 1996 – Kroll, 1996 – Liu et al., 2005 – Evsukoff et al., 2000 – Carrasco et al., 2004).

Sala et al. (Sala et al., 2005) notices that "Higher decision levels in process control also use rule bases for decision support. Supervision, diagnosis and condition monitoring are examples of successful application domains for fuzzy reasoning strategies".

In our work, unlike these approaches, fuzzy logic is used to design the parametric model.

In all cases, for the model-based approaches, the quality of the fault detection and isolation depends on the quality of the model.

It is possible to improve the model identification by implementing an original method based on a parameters adjustment by using a Fuzzy Forgetting Factor (FFF) (Lafont et al., 2005).

The idea, in this study, is to use the variations of the fuzzy forgetting factors for the fault detection and isolation.

Thus, we propose an original method based on a fuzzy adaptation of the parameter adjustments by introducing a fuzzy forgetting factor.

From these factors (one by output), we can generate residues for the fault detection and isolation. A numerical example, with several types of sensors defaults (the bias and the calibration default), is presented to show the performances of this method.

2 A NEW APPROACH: THE "FUZZY FORGETTING FACTOR" METHOD

In this section, after having presented the classical approach for the on-line identification, we present a new method of adaptation based on the fuzzy forgetting factor variation.

We consider a non-linear and non-stationary systems modeling. Consequently, an on-line adaptation is necessary to obtain a valid model capable of describing the process and allowing to realize an adaptive command (Fink et al., 2000). A common technique for estimating the unknown parameters is the Recursive Least Squares algorithm with forgetting factor (Campi, 1994 – Uhl, 2005 – Trabelsi et al., 2004).

At each moment *k*, we obtain a model, such as:

$$y(k+1) = A(k).y(k) + B(k).u(k) \tag{1}$$

with y the outputs vector and u the command vector,

$$\varphi(k) = \left(y(k)\, u(k)\right)^T \tag{2}$$

$$\hat{y}(k+1) = \hat{\theta}^T(k).\varphi(k) \tag{3}$$

$$\hat{\theta}(k+1) = \hat{\theta}(k) + m(k+1).P(k).\varphi^T(k).\varepsilon(k+1) \tag{4}$$

$$\varepsilon(k+1) = y(k+1) - \hat{y}(k+1) \tag{5}$$

$$P(k+1) = \frac{1}{\lambda(k)}\left[P(k) - \frac{P(k).\varphi(k).\varphi^T(k).P(k)}{\lambda(k) + \varphi^T(k).P(k).\varphi(k)}\right] \tag{6}$$

with $\hat{\theta}(k)$ the estimated parameters vector (initialized with the least-squares algorithm), $.\varphi(k)$ the regression vector, $\varepsilon(k+1)$ the a-posteriori error, $P(k)$ the gain matrix of regular adaptation and $\lambda(k)$ the forgetting factor.

If the process is slightly excited, the gain matrix $P(k)$ increases like an exponential (Slama-Belkhodja and de Fornel, 1985). To avoid this problem, and the drift of parameters, a measure $m(k)$ is introduced as:

$$m(k+1) = 1 \; if \left|\frac{u(k+1) - u(k)}{u_{\max}}\right| > S_u \tag{7}$$

$$\text{or } if \left|\frac{y(k+1) - \hat{\theta}^T(k).\varphi(k)}{y_n}\right| > S_y \tag{8}$$

$$m(k+1) = 0 \; if \left|\frac{u(k+1) - u(k)}{u_{\max}}\right| < S_u \tag{9}$$

$$\text{and } if \left|\frac{y(k+1) - \hat{\theta}^T(k).\varphi(k)}{y_n}\right| < S_y \tag{10}$$

with y_n the nominal value of y.

The adaptation is suspended as soon as the input becomes practically constant and/or as soon as the output y reaches a predefined tolerance area from the thresholds S_y and/or S_u. In the opposite case, and/or when a disturbance is detected on the input, the adaptation resumes with $m(k) = 1$.

The adaptation gain can be interpreted like a measurement of the parametric error. When an initial estimation of the parameters is available, the initial gain matrix is:

$$P(0) = G.I \tag{11}$$

With $G << 1$ or $Trace < 1$ and I : identity matrix.

We choose as initial values:

$$P(0) = \begin{bmatrix} 0.1 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0.1 \end{bmatrix} \tag{12}$$

and

$$\lambda(0) = 0.96 \tag{13}$$

2.1 Methods of the Forgetting Factor Variation

The considered class of the system imposes to use a method with a variable forgetting factor in order to take into account the non-stationarity of the process.

Generally, the adaptation of a model is obtained

by using a Recursive Least Squares algorithm with forgetting factor. The forgetting factor can be constant or variable.

There are different classical methods of the forgetting factor variation as, for example, the exponential forgetting factor. The variation of λ is defined as:

$$\lambda(k+1) = \lambda_0 . \lambda(k) + (1 - \lambda_0) \quad (14)$$

$$\text{where } 0 < \lambda_0 < 1 \quad (15)$$

with the typical values:

$$\lambda_0 = 0{,}95...0{,}99; \lambda(0) = 0{,}95...0{,}99 \quad (16)$$

This method consists in increasing λ to 1 rapidly.

Andersson proposes to modify the gain matrix $P(k+1)$ of the Recursive Least Squares algorithm to improve the model (Andersson, 1985). This method introduces an Adaptive forgetting Factor through Multiple Models (AFMM) in considering the RLS algorithm as a special case of the Kalman filter. $\hat{\theta}(k+1)$ is approximated with a sum of many Gaussian density functions. Moreover, when the process is subjected to jumps, this method enables us to reduce the importance of the gain matrix $P(k+1)$ in adjusting a parameter.

A new identification algorithm, inspired by these two methods (exponential and Andersson), is proposed. This approach presents a Fuzzy Forgetting Factor (Lafont et al., 2005).

2.2 The Proposed Approach

We use fuzzy logic to modify the forgetting factor in an automatic and optimal way (Jager, 1995). Thus, we have defined a «fuzzy box» of Mamdani type by using the following variables: $\lambda(k)$ and $\Delta\varepsilon(k)$ in input and $\lambda(k+1)$ in output (Figure 1).

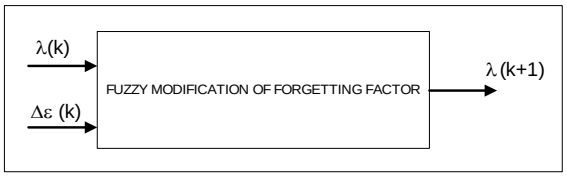


Figure 1: Fuzzy box.

$\Delta\varepsilon(k)$ represents the variation of the mean error on the N last samples:

$$\Delta\varepsilon(k) = \frac{1}{N} \sum_{j=k-N+1}^{k} (\varepsilon(j) - \varepsilon(j-1)) \quad (17)$$

$\Delta\varepsilon(k)$ had been defined with three membership functions: one for the negative error, one for the null error and one for the positive error (Figure 2). A study of observed process allows to determine the values: $\{-\eta_{\max}; -\eta_{\min}; \eta_{\min}; \eta_{\max}\}$.

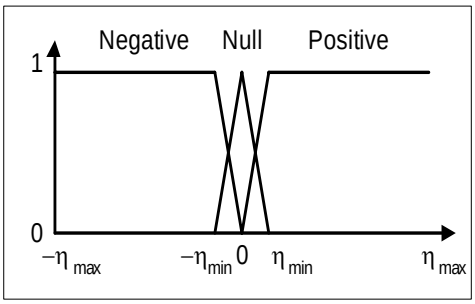


Figure 2: Fuzzyfication of the error variation.

The membership functions of the input $\lambda(k)$ and the output $\lambda(k+1)$ are identical (Figure 3).

According to the application, the bounds [0.8 ; 1] can be reduced.

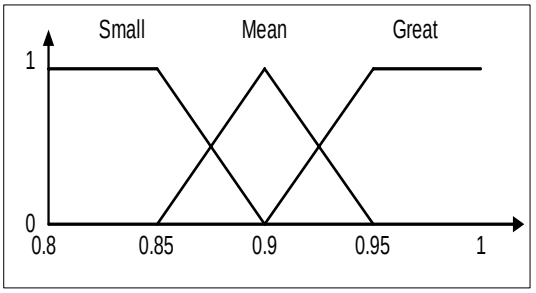


Figure 3: Fuzzyfication of the lambda.

The inference rules are based on the variation method of the exponential forgetting factor. In this case, the forgetting factor must be maximum when the modeling of the system is correct (small error variation). Also, we have been inspired by Andersson's work. When there is an important non-stationarity, the forgetting factor must decrease.

If $\lambda(k)$ is $F^1_{n_\lambda}$ and $\Delta\varepsilon(k)$ is $F^2_{n_\varepsilon}$ then $\lambda(k+1)$ is $F^3_{n'_\lambda}$, where $F^1_{n_\lambda} \in \{F^1_1, F^1_2, F^1_3\}$ is the set of membership functions of the input variable $\lambda(k)$, $F^2_{n_\varepsilon} \in \{F^2_1, F^2_2, F^2_3\}$ is the set of membership

functions of the input variable $\Delta\varepsilon(k)$ and $F^3_{n'_\lambda} \in \{F^3_1, F^3_2, F^3_3\}$ is the set of membership functions of $\lambda(k+1)$.

The rules for the output $\lambda(k+1)$ are defined in table 1.

Table 1: Rules for the variation of the forgetting factor.

$\Delta\varepsilon(k) \setminus \lambda(k)$	Small	Mean	Great
Negative	Small	Mean	Great
Null	Mean	Great	Great
Positive	Small	Small	Mean

The inference method is based on the max-min and the defuzzification is the centre of gravity.

$$\mu(z) = \max\left(\min\left(\min\left(\mu_{F^1_{n_\lambda}}(\upsilon), \mu_{F^2_{n_\varepsilon}}(\nu)\right)\right), \mu_{F^3_{n'_\lambda}}(z)\right) \quad (18)$$

With $n_\lambda = 1$ to 3, $n_\varepsilon = 1$ to 3 and $n'_\lambda = 1$ to 3.

$$\lambda(k+1) = \frac{\int \mu(z).z.dz}{\int \mu(z).dz} \quad (19)$$

The number of forgetting factors is equal to the number of model outputs.

3 GENERATION OF RESIDUES AND DECISION-MAKING

3.1 Classical Method

The residuals are analytical redundancy generated measurements representing the difference between the observed and the expected system behaviour. When a fault occurs, the residual signal allows to evaluate the difference with the normal operating conditions.

The residuals are processed and examined under certain decision rules to determine the change of the system status. Thus, the fault is detected, isolated (to distinguish the abnormal behaviours and determine the faulty component) and identified (to characterize the duration of the default and the amplitude in order to deduce its severity).

A threshold between the outputs of the system and the estimated outputs is chosen in order to proceed to the decision-making.

The residues $r_j = \left|\left(y_j - \hat{y}_j\right)\right|$ are calculated to estimate the case where there is no failure and the case of sensor default. A threshold t is taken: if $r_j \le t$ *then* $r_j = 0$.

At each instant k, the different r_j are checked in order to establish a diagnosis.

3.2 Approach with Fuzzy Lambda

Our method uses the fuzzy lambda to detect and isolate a default on a sensor. For the MIMO system, the algorithm generates one lambda for each output.

Let λ_j, with $j = 1$ to n, n : number of outputs.

The residues $r'_j = 1 - \lambda_j$ are calculated to estimate the case where there is no failure and the case of sensor default. A threshold t' is taken: if $r'_j \le t'$ *then* $r'_j = 0$.

At each instant k, the different r'_j are checked in order to establish a diagnosis as shown in table 2.

Table 2: Analysis of residues.

Analysis of residues	Diagnosis
$\forall\ j, r'_j = 0$	No failure
If $\exists$ *one and only one* $r'_j \neq 0$, $t' = index\{r', r'_j \neq 0\}$	Sensor default

4 APPLICATION

4.1 Benchmark Example: A Hydraulic Process (Jamouli, 2003)

The approach proposed previously has been validated on a benchmark: a hydraulic process. This system is a hydraulic process composed of three tanks (Figure 4). The objective of the regulation is to be able to have a constant volume of the fluid. The three tanks have the same section: S.

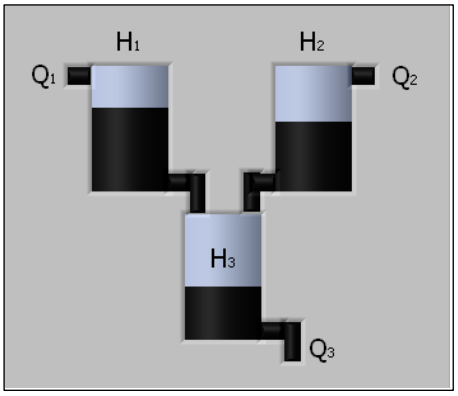


Figure 4: A hydraulic process.

The physical model of this system is obtained with the difference between the entering and outgoing flows which make evolve the level of each tank.

The state model is described by:

$$\begin{aligned} \dot{X} &= A.X + B.U \\ Y &= C.X + D.U \end{aligned} \tag{20}$$

$$\begin{aligned} &X = [h_1\ h_2\ h_3]^T, U = [q_1\ q_2]^T \\ &and\ Y = X \end{aligned} \tag{21}$$

The vector of outputs is the same as the state vector and, thus, the observation matrix C is an identity matrix with a size 3x3.

This system, considered as linear around a running point, has been identified in using an ARX structure. The discrete model is obtained by using a sample period equal to 0.68 seconds.

The model describes the dynamical behaviour of the system in terms of inputs/outputs variations around the running point $(U_0\ Y_0)$.

$$U_0 = (0.8\ \ 1)^T\ \ Y_0 = (400\ \ 300\ \ 200)^T$$

$$\begin{aligned} x(k+1) &= A_d.x(k) + B_d.u(k) \\ y(k) &= C_d.x(k) + D_d.u(k) + no(k) \end{aligned} \tag{22}$$

The sensors noise $no(k)$ considered is a normal distribution with mean zero and variance one.

This system is completely observable and controllable.

A quadratic linear control, associated to an integrator, enables to calculate the feedback gain matrix K from the minimization of the following cost function:

$$J = \frac{1}{2}.\sum_{k=0}^{N}\left(x^T(k).Q.x(k) + u^T(k).R.u(k)\right) \tag{23}$$

$$u(k) = -K.x(k) \tag{24}$$

As shown in section 2, for each output, a forgetting factor is assigned. λ_1, λ_2 and λ_3 vary independently in function of the error between the process outputs and the model outputs.

For this application, the values $\eta_{\min}$ and $\eta_{\max}$, described in section 2.2, are respectively 1.25 and 10. The model is adapted to follow the process behaviour.

4.2 Sensors Defaults

For the sensors, two types of defaults have been tested: the bias and the calibration default. The simulation of the bias default has been carried out by substracting a constant value β from the real value: for example $h_{1\,real} = h_{1\,sensor} - \beta$.

The simulation of the calibration default is obtained by multiplying the real value by a coefficient γ : for example $h_{1\,real} = h_{1\,sensor} * \gamma$.

The environment of the supervision enables to see the good detection of defaults. As soon as a failure is detected, the algorithm stops and indicates which sensor has a default (Figure 5). The physical model is represented by the dotted line curve and the parametric model by the solid line curve. For this example, the default is simulated, at sample 10, on the sensor h_1. The diagnosis is depicted by a circle on figure 5. The algorithm has detected the default at sample 12.

We have simulated the classical method and our approach with the bias default and the calibration default for the three sensors (h_1, h_2, h_3). To compare these two methods, we vary the values β and γ.

4.3 Results

In table 3 and table 4, we show the performances of the two methods. For this, we define a rate which is the percentage of detection on 100 tests.

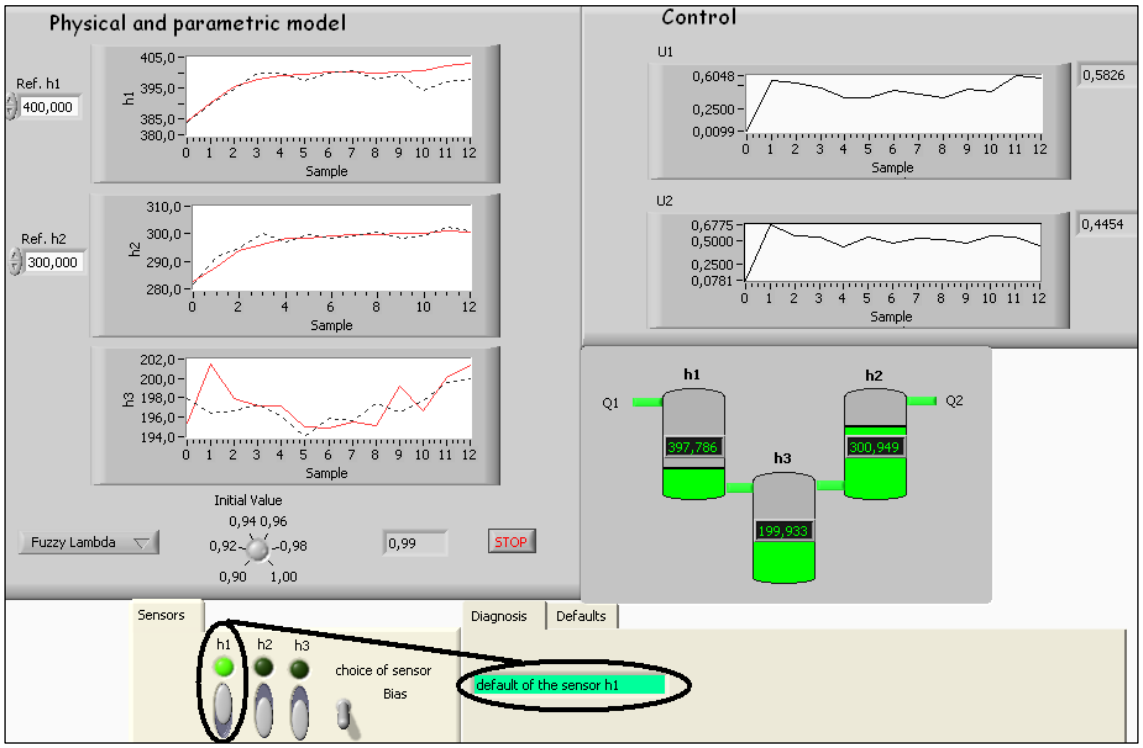


Figure 5: Supervision.

We can note that the fuzzy method gives better results. Indeed, when the default is weak (β <7 or γ >0.97), the rate of detection is more important.

On the other hand, the results are similar. To improve the detection with the classical method, the threshold t could be decreased but that implies an important rate of false alarm. Indeed, if the threshold is weaker than the importance of the noise, the algorithm stops in an inopportune way.

Table 3: Rate of detection for the bias default.

Bias default Rate of detection in percentage		β				
		4	5	6	7	8
Classical method Threshold: t= 5.5	h_1	6	42	88	98	100
	h_2	6	42	80	100	100
	h_3	22	42	80	90	100
Fuzzy method Threshold: t'= 0.1	h_1	64	84	100	100	100
	h_2	76	92	98	100	100
	h_3	86	92	98	100	100

Table 4: Rate of detection for the calibration default.

Calibration default Rate of detection in percentage		γ			
		0.99	0.98	0.97	0.96
Classical method Threshold: t= 5.5	h_1	24	100	100	100
	h_2	2	84	100	100
	h_3	0	8	80	98
Fuzzy method Threshold: t'= 0.1	h_1	70	100	100	100
	h_2	48	100	100	100
	h_3	36	68	92	100

4.4 Sensitivity to the Measure Noise

The measure noise has a great significance on the fault detection. The presented values are the minimal values which the method can detect.

In the case where the measure noise is more important, these results can be upgraded by modifying the values η_{min} and η_{max} defined in section 2.2. If the measure noise is very large, it is necessary to increase these initial values. By doing that, a tolerance compared with the noise is admitted. A compromise should be found between

the noise level and the variation of $\eta_{\min}$ and $\eta_{\max}$. Indeed, the algorithm can detect a false alarm.

5 CONCLUSIONS

This paper presents an original method of model-based diagnosis with a fuzzy parametric approach. This method is applicable to all non-linear MIMO systems for which the knowledge of the physical model is not required. We define a Fuzzy Forgetting Factor which allows to improve the estimation of model parameters, and to detect and isolate several types of faults. Thus, the fuzzy adaptation of the forgetting factors is used to detect and isolate the sensor fault. The results are illustrated by a benchmark system (a hydraulic process) and comparisons between the classical method and this method is depicted in table 3 and table 4.

The method is efficient to detect and isolate only one sensor default at the same moment. The proposed approach is able to detect faults which correspond to a bias and a calibration default for a sensor.

A possible extension would be to determine the values $\eta_{\min}$ and $\eta_{\max}$, described in section 2.2, in an automatic way according to the sensor noise.

Moreover, it would be interesting to develop the FFF method for the actuator defaults.

REFERENCES

Andersson, P., 1985. Adaptive forgetting in recursive identification through multiple. *In. Proc. Int. J. Control, pp. 1175-1193.*

Campi, M., 1994. Performance of RLS Identification Algorithms with Forgetting Factor: A Φ-Mixing Approach, *Journal of Mathematical Systems, Estimation, and Control, Vol. 4, N° 3, pp. 1-25.*

Carrasco, E. F., Rodriguez, J., Punal, A., Roca, E., Lema, J. M., 2004. Diagnosis of acidification states in an anaerobic wastewater treatment plant using a fuzzy-based expert system, *Control Engineering Practice, 12, pp. 59-64.*

Evsukoff, A., Gentil, S., Montmain, J., 2000. Fuzzy reasoning in co-operative supervision systems, *Control Engineering Practice, 8, pp. 389-407.*

Fink, A., Fischer, M., Nelles, O., 2000. Supervision of Non-linear Adaptive Controllers Based on Fuzzy Models, *Control Engineering Practice, 8(10), pp. 1093-1105.*

Isermann, R., 1984. Process fault detection based on modelling and estimation methods – A survey, *Automatica,vol. 20, n°4, pp. 387-404.*

Isermann, R., 1997. Supervision, fault-detection and fault-diagnosis methods-Advanced methods and applications, *Proc. Of the IMEKO world congress, New Measurements – Challenges and Visions, Tampere, Finland, vol. 1, n°4, pp. 1-28.*

Isermann, R., 2005. Model-based fault detection and diagnosis - Status and applications, *Annual Reviews in Control, Elsevier Ltd., pp. 71-85, Vol. 28, No. 1.*

Jager, R., 1995. Fuzzy Logic in Control, *Thesis Technische Universiteit Delft, ISBN 90-9008318-9.*

Jamouli, H., 2003. Génération de résidus directionnels pour le diagnostic des systèmes linéaires stochastiques et la commande tolérante aux fautes, *Thesis, University Henri Poincaré, Nancy 1.*

Kroll, A., 1996. Identification of functional fuzzy models using multidimensional reference fuzzy sets, *Fuzzy Sets & Systems, vol. 8, pp. 149-158.*

Lafont, F., Balmat, J. F., Taurines, M., 2005. Fuzzy forgetting factor for system identification, *Third International Conference on Systems, Signals & Devices, Volume 1, Systems analysis & Automatic Control, Sousse, Tunisia, March 21-24.*

Liu, G., Toncich, D. J., Harvey, E. C., Yuan, F., 2005. Diagnostic technique for laser micromachining of multi-layer thin films, *International Journal of Machine Tools & Manufacture, 45, pp. 583-589.*

Maquin, D., 1997. Diagnostic à base de modèles des systèmes technologiques, *Mémoire d'Habilitation à Diriger des Recherches, Institut National Polytechnique de Lorraine.*

Noura, H., 2002. Méthodes d'accommodation aux défauts: théorie et application, *Mémoire d'Habilitation à Diriger des Recherches, University Henri Poincaré, Nancy 1.*

Querelle, R., Mary, R., Kiupel, N., Frank, P. M., 1996. Use of qualitative modelling and fuzzy clustering for fault diagnosis, *Proc. of world Automation Congress WAC'96, Montpellier, France, vol. 5, n°4, pp. 527-532.*

Ripoll, P., 1999. Conception d'un système de diagnostic flou appliqué au moteur automobile, *Thesis, the University of Savoie.*

Sala, A., Guerra, T. M., Babuska, R., 2005. Perspectives of fuzzy systems and control, *Fuzzy Sets & Systems, 156, pp. 432-444.*

Slama-Belkhodja, I., de Fornel, B., 1996. Commande adaptative d'une machine asynchrone, *J. Phys. III, Vol. 6, pp. 779-796.*

Szederkényi, G., 1998. Model-Based Fault Detection of Heat Exchangers, *department of Applied Computer Science University of Veszprém.*

Trabelsi, A., Lafont, F., Kamoun, M., Enéa, G., 2004. Identification of non-linear multi-variable systems by adaptive Fuzzy Takagi-Sugeno model, *International Journal of Computational Cognition, ISBN 1542-5908, vol. 2, n° 3, pp. 137-153.*

Uhl, T., 2005. Identification of modal parameters for non-stationary mechanical systems, *Arch Appl Mech, 74, pp. 878-889.*

BREAKING ACCESSIBILITY BARRIERS
Computational Intelligence in Music Processing for Blind People

Wladyslaw Homenda
Faculty of Mathematics and Information Science
Warsaw University of Technology, pl. Politechniki 1, 00-661 Warsaw, Poland
homenda@mini.pw.edu.pl

Keywords: Knowledge processing, music representation, music processing, accessibility, blindness.

Abstract: A discussion on involvement of knowledge based methods in implementation of user friendly computer programs for disabled people is the goal of this paper. The paper presents a concept of a computer program that is aimed to aid blind people dealing with music and music notation. The concept is solely based on computational intelligence methods involved in implementation of the computer program. The program is build around two research fields: information acquisition and knowledge representation and processing which are still research and technology challenges. Information acquisition module is used for recognizing printed music notation and storing acquired information in computer memory. This module is a kind of the paper-to-memory data flow technology. Acquired music information stored in computer memory is then subjected to mining implicit relations between music data, to creating a space of music information and then to manipulating music information. Storing and manipulating music information is firmly based on knowledge processing methods. The program described in this paper involves techniques of pattern recognition and knowledge representation as well as contemporary programming technologies. It is designed for blind people: music teachers, students, hobbyists, musicians.

1 INTRODUCTION

In this paper we attempt to study an application of methods of computational intelligence in the real life computer program that is supposed to handle music information and to provide an access for disabled people, for blind people in our case. The term *computational intelligence*, though widely used by computer researchers, has neither a common definition nor it is uniquely understood by the academic community. However, it is not our aim to provoke a discussion on what artificial intelligence is and which methods it does embed. Instead, we rather use the term in a common sense. In this sense intuitively understood knowledge representation and processing is a main feature of it. Enormous development of computer hardware over past decades has enabled bringing computers as tools interacting with human partners in an intelligent way. This required, of course, the use of methods that firmly belong to the domain of computational intelligence and widely apply knowledge processing.

Allowing disabled people to use computer facilities is an important social aspect of software and hardware development. Disabled people are faced problems specific to their infirmities. Such problems have been considered by hardware and software producers. Most important operating systems include integrated accessibility options and technologies. For instance, Microsoft Windows includes Active Accessibility techniques, Apple MacOS has Universal Access tools, Linux brings Gnome Assistive Technology. These technologies support disabled people and, also, provide tools for programmers. They also stimulate software producers to support accessibility options in created software. Specifically, if a computer program satisfies necessary cooperation criteria with a given accessibility technology, it becomes useful for disabled people.

In the age of information revolution development of software tools for disabled people is far inadequate to necessities. The concept of music processing support with a computer program dedicated to blind people is aimed to fill in a gap between requirements and tools available. Bringing accessibility technology to blind people is usually based on computational intelligence methods such as pattern recognition and knowledge representation. Music processing com-

puter program discussed in this paper, which is intended to contribute in breaking the accessibility barrier, is solely based on both fields. Pattern recognition is applied in music notation recognition. Knowledge representation and processing is used in music information storage and processing.

1.1 Notes on Accessibility for Blind People

The population of blind people is estimated to up to 20 millions. Blindness, one of most important disabilities, makes suffering people unable to use ordinary computing facilities. They need dedicated hardware and, what is even more important, dedicated software. In this Section our interest is focused on accessibility options for blind people that are available in programming environments and computer systems.

An important standard of accessibility options for disabled people is provided by IBM Corporation. This standard is common for all kinds of personal computers and operating systems. The fundamental technique, which must be applied in blind people aimed software, relies on assigning all program functions to keyboard. Blind people do not use mouse or other pointing devices, thus mouse functionality must also be assigned to keyboard. This requirement allows blind user to learn keyboard shortcuts which activates any function of the program (moreover, keyboard shortcuts often allow people with good eyesight to master software faster then in case of mouse usage). For instance, Drag and Drop, the typical mouse operation, should be available from keyboard. Of course, keyboard action perhaps will be entirely different then mouse action, but results must be the same in both cases. Concluding, well design computer program must allow launching menus and context menus, must give access to all menu options, toolbars, must allow launching dialog boxes and give access to all their elements like buttons, static and active text elements, etc. These constraints need careful design of program interface. Ordering of dialog box elements which are switched by keyboard actions is an example of such a requirement.

Another important factor is related to restrictions estimated for non disabled users. For instance, if application limits time of an action, e.g. waiting time for an answer, it should be more tolerant for blind people since they need more time to prepare and input required information.

Application's design must consider accessibility options provided by the operating system in order to avoid conflicts with standard options of the system. It also should follow standards of operating system's accessibility method. An application for blind people should provide conflict free cooperation with screen readers. It must provide easy-to-learn keyboard interface duplicating operations indicated by pointing devices.

Braille display is the basic hardware element of computer peripherals being a communicator between blind man and computer. It plays roles of usual screen, which is useless for blind people, and of control element allowing for a change of screen focus, i.e. the place of text reading. Braille display also communicates caret placement and text selection.

Braille printer is another hardware tool dedicated to blind people. Since ordinary printing is useless for blind people, Braille printer punches information on special paper sheet in form of the Braille alphabet of six-dots combinations. Punched documents play the same role for blind people as ordinary printed documents for people with good eyesight.

Screen reader is the basic software for blind people. Screen reader is the program which is run in background and which captures content of an active window or an element of a dialog box and communicates it as synthesized speech. Screen reader also keeps control over Braille display communicating information that is simultaneously spoken.

Braille editors and converters are groups of computer programs giving blind people access to computers. Braille editors allow for editing and control over documents structure and contents. Converters translate ordinary documents to Braille form and oppositely.

1.2 Notes on Software Development for Blind People

Computers become widely used by disabled people including blind people. It is very important for blind people to provide individuals with technologies of easy transfer of information from one source to another. Reading a book becomes now as easy for blind human being as for someone with good eyesight. Blind person can use a kind of scanning equipment with a speech synthesizer and, in this way, may have a book read by a computer or even displayed at a Braille display. Advances in speech processing allow for converting printed text into spoken information. On the other hand, Braille displays range from linear text display to two dimensional Braille graphic windows with a kind of gray scale imaging. Such tools allow for a kind of reading or seeing and also for editing of texts and graphic information.

Text processing technologies for blind people are now available. Text readers, though still very ex-

pensive and not perfect yet, becomes slowly a standard tool of blind beings. Optical character recognition, the heart of text readers, is now well developed technology with almost 100% recognition efficiency. This perfect technology allows for construction of well working text readers. Also, current level of development of speech synthesis technology allows for acoustic communicating of a recognized text. Having text's information communicated, it is easy to provide tools for text editing. Such editing tools usually use a standard keyboard as input device.

Text processing technologies are rather exceptions among other types of information processing for blind people. Neither more complicated document analysis, nor other types of information is easily available. Such areas as, for instance, recognition of printed music, of handwritten text and handwritten music, of geographical maps, etc. still raise challenges in theory and practice. Two main reasons make that software and equipment in such areas is not developed for blind people as intensively as for good eyesight ones. The first reason is objective - technologies such as geographical maps recognition, scanning different forms of documents, recognizing music notation are still not well developed. The second reason is more subjective and is obvious in commercial world of software publishers - investment in such areas scarcely brings profit.

2 ACQUIRING MUSIC INFORMATION

Any music processing system must be supplied with music information. Manual inputs of music symbols are the easiest and typical source of music processing systems. Such inputs could be split in two categories. One category includes inputs form - roughly speaking - computer keyboard (or similar computer peripheral). Such input is usually linked to music notation editor, so it affects computer representation of music notation. Another category is related to electronic instruments. Such input usually produce MIDI commands which are captured by a computer program and collected as MIDI file representing live performance of music.

Besides manual inputs we can distinguish inputs automatically converted to human readable music formats. The two most important inputs of automatic conversion of captured information are automatic music notation recognition which is known as Optical Music Recognition technology and audio music recognition known as Digital Music Recognition technology. In this paper we discuss basics of automatic music notation recognition as a source of input information feeding music processing computer system.

2.1 Optical Music Recognition

Printed music notation is scanned to get image files in TIFF or similar format. Then, OMR technology converts music notation to the internal format of computer system of music processing. The structure of automated notation recognition process has two distinguishable stages: location of staves and other components of music notation and recognition of music symbols. The first stage is supplemented by detecting score structure, i.e. by detecting staves, barlines and then systems and systems' structure and detecting other components of music notation like title, composer name, etc. The second stage is aimed on finding placement and classifying symbols of music notation. The step of finding placement of music notation symbols, also called segmentation, must obviously precede the step of classification of music notation symbols. However, both steps segmentation and classification often interlace: finding and classifying satellite symbols often follows classification of main symbols.

2.1.1 Staff Lines and Systems Location

Music score is a collection of staves which are printed on sheets of paper, c.f. (Homenda, 2002). Staves are containers to be filled in with music symbols. Stave(s) filled in with music symbols describe a part played by a music instrument. Thus, stave assigned to one instrument is often called a part. A part of one instrument is described by one stave (flute, violin, cello, etc.) or more staves (two staves for piano, three staves for organ).

Staff lines location is the first stage of music notation recognition. Staff lines are the most characteristic elements of music notation. They seem to be easily found on a page of music notation. However, in real images staff lines are distorted raising difficulties in automatic positioning. Scanned image of a sheet of music is often skewed, staff line thickness differs for different lines and different parts of a stave, staff lines are not equidistant and are often curved, especially in both endings of the stave, staves may have different sizes, etc., c.f. (Homenda, 1996; Homenda, 2002) and Figure 1.

Having staves on page located, the task of system detection is performed. Let us recall that the term system (at a page of music notation) is used in the meaning of all staves performed simultaneously and joined together by beginning barline. Inside and ending bar-

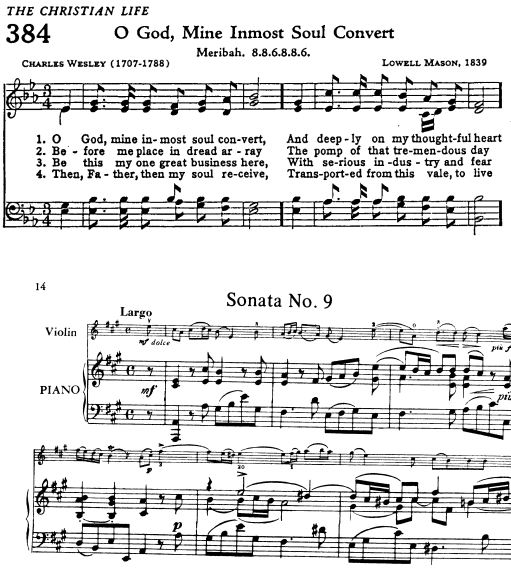


Figure 1: Examples of real notations subjected to recognition.

lines define system's structure. Thus, detection of systems and systems' structure relies on finding barlines.

2.1.2 Score Structure Analysis

Sometimes one stave includes parts of two instruments, e.g. simultaneous notation for flute and oboe or soprano and alto as well as tenor and bass. All staves, which include parts played simultaneously, are organized in systems. In real music scores systems are often irregular, parts which not play may be missing.

Each piece of music is split into measures which are rhythmic, (i.e. time) units defined by time signature. Measures are separated from each other by barlines.

The task of score structure analysis is to locate staves, group them into systems and then link respective parts in consecutive systems. Location of barlines depicts measures, their analysis split systems into group of parts and defines repetitions.

2.1.3 Music Symbol Recognition

Two important problems are raised by symbol recognition task: locating and classifying symbols. Due to irregular structure of music notation, the task of finding symbol placement decides about final symbol recognition result. Symbol classification could not give good results if symbol location is not well done. Thus, both tasks are equally important in music symbols recognition.

Figure 2: Printed symbols of music notation - distortions, variety of fonts.

Since no universal music font exits, c.f. Figure 1, symbols of one class may have different forms. Also size of individual symbols does not keep fixed proportions. Even the same symbols may have different sizes in one score. Besides usual noise (printing defects, careless scanning) extra noise is generated by staff and ledger lines, densely packed symbols, conflicting placement of other symbols, etc.

A wide range of methods are applied in music symbol recognition: neural networks, statistical pattern recognition, clustering, classification trees, etc., c.f. (Bainbridge and Bell, 2001; Carter and Bacon, 1992; Fujinaga, 2001; Homenda and Mossakowski, 2004; McPherson, 2002). Classifiers are usually applied to a set of features representing processed symbols, c.f. (Homenda and Mossakowski, 2004). In next section we present application of neural networks as example classifier.

2.2 Neural Networks as Symbol Classifier

Having understood the computational principles of massively parallel interconnected simple neural processors, we may put them to good use in the design of practical systems. But neurocomputing architectures are successfully applicable to many real life problems. The single or multilayer fully connected feedforward or feedback networks can be used for character recognition, c.f. (Homenda and Luckner, 2004).

Experimental tests were targeted on classification of quarter, eight and sixteen rests, sharps, flats and naturals, c.f. Figure 2 for examples music symbols. To reduce dimensionality of the problem, the images were transformed to a space of 35 features. The method applied in feature construction was the simplest one, i.e. they were created by hand based on understanding of the problem being tackled. The list of features included the following parameters computed for bounding box of a symbol and for four quarters of bounding box spawned by symmetry axes of the bounding box:

- mean value of vertical projection,

- slope angle of a line approximating vertical projection,
- slope angle of a line approximating histogram of vertical projection;
- general horizontal moment m_{10},
- general vertical moment m_{01},
- general mixed moment m_{11}.

The following classifiers were utilized: backpropagation perceptron, feedforward counterpropagation maximum input network and feedforward counterpropagation closest weights network. An architecture of neural network is denoted by a triple input - hidden - output which identifies the numbers of neurons in input, hidden and output layers, respectively, and does not include bias inputs in input and hidden layers. The classification rate for three symbols on music notation: flats, sharps and naturals ranges between 89% and 99 9% , c.f. (Homenda and Mossakowski, 2004). Classifier applied: backpropagation perceptron, feedforward counterpropagation maximum input network and feedforward counterpropagation closest weights network. An architecture of neural network is denoted by a triple *input - hidden - output* which identifies the numbers of neurons in input, hidden and output layers, respectively, and does not include bias inputs in input and hidden layers.

3 REPRESENTING MUSIC INFORMATION

Acquired knowledge has to be represented and stored in a format *understandable* by the computer brain, i.e. by a computer program - this is a fundamental observation and it will be exploited as a subject of discussion in this section. Of course, a computer program cannot work without low level support - it uses a processor, memory, peripherals, etc., but they are nothing more than only primitive electronic tools and so they are not interesting from our point of view. Processing of such an acquired image of the paper document is a clue to the paper-to-memory data transfer and it is successfully solved for selected tasks, c.f. OCR technology. However, documents that are more complicated structurally than linear (printed) texts raise the problem of data aggregation in order to form structured space of information. Such documents raise the problem of acquiring of implicit information/knowledge that could be concluded from the relationships between information units. Documents containing graphics, maps, technical drawings, music notation, mathematical formulas, etc. can illustrate these aspects of difficulties of paper-to-computer-memory data flow. They are research subjects and still raise a challenge for software producers.

Optical music recognition (OMR) is considered as an example of paper-to-computer-memory data flow. This specific area of interest forces specific methods applied in data processing, but in principle, gives a perspective on the merit of the subject of knowledge processing. Data flow starts from a raster image of music notation and ends with an electronic format representing the information expressed by a scanned document, i.e. by music notation in our case. Several stages of data mining and data aggregation convert the chaotic ocean of raster data into shells of structured information that, in effect, transfer structured data into its abstraction - music knowledge. This process is firmly based on the nature of music notation and music knowledge. The global structure of music notation has to be acquired and the local information fitting this global structure must also be recovered from low level data. The recognition process identifies structural entities like staves, group them into higher level objects like systems, than it links staves of sequential systems creating instrumental parts. Music notation symbols very rarely exist as standalone objects. They almost exclusively belong to structural entities: staves, systems, parts, etc. So that the mined symbols are poured into these prepared containers - structural objects, cf. (Bainbridge and Bell, 2001; Dannenberg and Bell, 1993; Homenda, 2002; Taube, 1993). Music notation is a two dimensional language in which the importance of the geometrical and logical relationships between its symbols may be compared to the importance of the symbols alone. This phenomenon requires that the process of music knowledge acquisition must also be aimed at recovering the implicit information represented by the geometrical and logical relationships between the symbols and then at storing the recovered implicit relationships in an appropriate format of knowledge representation.

There are open problems of information gaining and representation like, for instance, performance style, timbre, tone-coloring, feeling, etc. These kinds of information are neither supported by music notation, not could be derived in a reasoning process. Such kinds of information are more subjectively perceived rather than objectively described. The problem of definition, representation and processing of ”subjective kinds of information” seems to be very interesting from research and practical perspectives. Similarly, problems like, for instance, human way of reading of music notation may be important from the

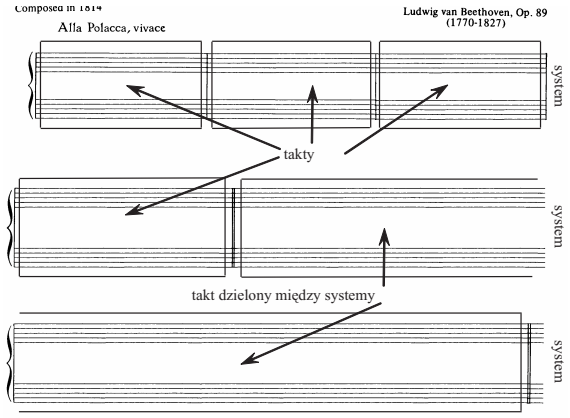


Figure 3: Structuring music notation - systems and measures.

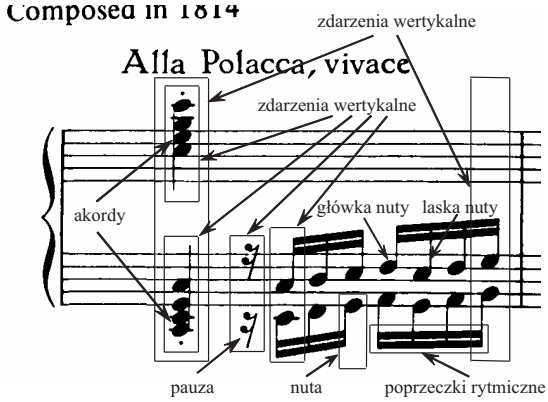


Figure 4: Structuring music notation - symbols, ensembles of symbols.

point of view of music score processing, cf. (Goolsby, 1994a; Goolsby, 1994b). Nevertheless, processing of such kinds of information does not fit framework of the paper and is not considered.

The process of paper-to-computer-memory music data flow is presented from the perspective of a paradigm of granular computing, cf. (Pedrycz, 2001). The low-level digitized data is an example of numeric data representation, operations on low-level data are numeric computing oriented. The transforming of a raster bitmap into compressed form, as e.g. run lengths of black and while pixels, is obviously a kind of numeric computing. It transfers data from its basic form to more compressed data. However, the next levels of the data aggregation hierarchy, e.g. finding the handles of horizontal lines, begins the process of data concentration that become embryonic knowledge units rather than more compressed data entities, cf. (Homenda, 2005).

3.1 Staves, Measures, Systems

Staves, systems, measures are basic concepts of music notation, cf. Figure 3. They define the structure of music notation and are considered as information quantities included into the data abstraction level of knowledge hierarchy. The following observations justify such a qualification.

A stave is an arrangement of parallel horizontal lines which together with the neighborhood are the locale for displaying musical symbols. It is a sort of vessel within a system into which musical symbols can be ”poured”. Music symbols, text and graphics that are displayed on it belong to one or more parts. Staves, though directly supported by low-level data, i.e. by a collection of black pixels, are complex geometrical shapes that represent units of abstract data. A knowledge unit describing a stave includes such geometrical information as the placement (vertical and horizontal) of its left and right ends, staff lines thickness, the distance between staff lines, skew factor, curvature, etc. Obviously, this is a complex quantity of data.

A system is a set of staves that are played in parallel; in printed music all of these staves are connected by a barline drawn through from one stave to next on their left end. Braces and/or brackets may be drawn in front of all or some of them.

A measure is usually a part of a system, sometimes a measure covers the whole system or is split between systems, cf. Figure 3. A measure is a unit of music identified by the time signature and rhythmic value of the music symbols of the measure. Thus, like in the above cases, a measure is also a concept of data abstraction level.

3.2 Notes, Chords, Vertical Events, Time Slices

Such symbols and concepts as notes, chords, vertical events, time slices are basic concepts of music notation, cf. Figure 4. They define the local meaning of music notation and are considered as information quantities included in the data abstraction level of the knowledge hierarchy. Below a description of selected music symbols and collections of symbols are described. Such a collection constitutes a unit of information that has common meaning for musician. These descriptions justify classification of symbols to in the data abstraction level of the knowledge hierarchy.

Note - a symbol of music notation - represents basically the tone of given time, pitch and duration. A

note may consist of only a notehead (a whole note) or also have a stem and may also have flag(s) or beam(s). The components of a note are information quantities created at the data concentration and the data aggregation stages of data aggregation process. This components linked in the concept of a note create an abstract unit of information that is considered as a more complex component of the data abstraction level of the information hierarchy.

A chord is composed of several notes of the same duration with noteheads linked to the same stem (this description does not extend to whole notes due to the absence of a stem for such notes). Thus, a chord is considered as data abstraction.

A vertical event is the notion by which a specific point in time is identified in the system. Musical symbols representing simultaneous events of the same system are logically grouped within the same vertical event. Common vertical events are built of notes and/or rests.

A time slice is the notion by which a specific point in time is identified in the score. A time slice is a concept grouping vertical events of the score specified by a given point in time. Music notation symbols in a music representation file are physically grouped by page and staff, so symbols belonging to a common time slice may be physically separated in the file. In most cases this is time slice is split between separated parts for the scores of part type, i.e. for the scores with parts of each performer separated each of other. Since barline can be seen as a time reference point, time slices can be synchronized based on barline time reference points. This fact allows for localization of recognition timing errors to one measure and might be applied in error checking routine.

4 BRAILLE SCORE

Braille Score is a project developed and aimed on blind people. Building integrated music processing computer program directed to a broad range of blind people is the key aim of Braille Score, c.f. (Moniuszko, 2006). The program is mastered by a man. Both the man and computer program create an integrated system. The structure of the system is outlined in Figure 5.

The system would act in such fields as:

- creating scores from scratch,
- capturing existing music printings and converting them to electronic version,
- converting music to Braille and printing it automatically in this form,

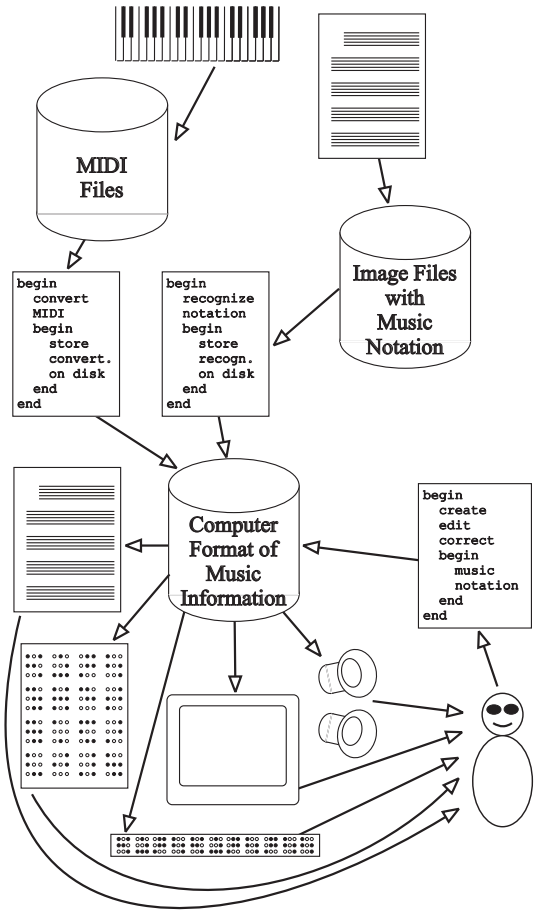


Figure 5: The structure of Braille Score.

- processing music: transposing music to different keys, extracting parts from given score, creating a score from given parts,
- creating and storing own compositions and instrumentations of musicians,
- a teacher's tool to prepare teaching materials,
- a pupil's tool to create their own music scores from scratch or adapt acquired music,
- a hobby tool.

4.1 User Interface Extensions for Blind People

Braille Score is addressed to blind people, c.f (Moniuszko, 2006). Its user interface extensions allow blind user to master the program and to perform operations on music information. Ability to read, edit and print music information in Braille format is the most important feature of Braille Score. Blind user is provided the following elements of interface: Braille

notation editor, keyboard as input tool, sound communicator.

Blind people do not use pointing devices. In consequence, all input functions usually performed with mouse must be mapped to computer keyboard. Massive communication with usage of keyboard requires careful design of interface mapping to keyboard, c.f. (Moniuszko, 2006).

Blind user usually do not know printed music notation. Their perception of music notation is based on Braille music notation format presented at Braille display or punched sheet of paper, c.f. (Krolick, 1998). In such circumstances music information editing must be done on Braille music notation format. Since typical Braille display is only used as output device, such editing is usually done with keyboard as input device. In Braille Score Braille representation of music is converted online to internal representation and displayed in the form of music notation in usual form. This transparency will allow for controlling correctness and consistency of Braille representation, c.f. (Moniuszko, 2006).

Sound information is of height importance for blind user of computer program. Wide spectrum of visual information displayed on a screen for user with good eyesight could be replaced by sound information. Braille Score provides sound information of two types. The first type of sound information collaborates with screen readers, computer programs dedicated to blind people. Screen readers could read contents of a display and communicate it to user in the form of synthesized speech. This type of communication is supported by contemporary programming environments. For this purpose Braille Score uses tools provided by Microsoft .NET programming environment. The second type of sound information is based on own Braille Score tools. Braille Score has embedded mechanism of sound announcements based on its own library of recorded utterances.

5 CONCLUSIONS

The aim of this paper is a discussion on involvement of computational intelligence methods in implementation of user friendly computer programs focused on disabled people. In te paper we describe a concept of Braille Score the specialized computer program which should help blind people to deal with music and music notation. The use of computational intelligence tolls can improve the program part devoted to recognition and processing of music notation. The first results with Braille Score show its to be a practical and useful tool.

REFERENCES

Bainbridge, D. and Bell, T. (2001). The challenge of optical music recognition. *Computers and the Humanities*, 35:95–121.

Carter, N. P. and Bacon, R. A. (1992). *Automatic Recognition of Printed Music*, pages 456–465. in: Structured Document Analysis, Analysis, H.S.Baird, H.Bunke, K.Yamamoto (Eds). Springer Verlag.

Dannenberg, R. and Bell, T. (1993). Music representation issues, techniques, and systems. *Computer Music Journal*, 17(3):20–30.

Fujinaga, I. (2001). Adaptive optical music recognition. In *16th Inter. Congress of the Inter. Musicological Society, Oxford*.

Goolsby, T. W. (1994a). Eye movement in music reading: Effects of reading ability, notational complexity, and encounters. *Music Perception*, 12(1):77–96.

Goolsby, T. W. (1994b). Profiles of processing: Eye movements during sightreading. *Music Perception*, 12(1):97–123.

Homenda, W. (1996). Automatic recognition of printed music and its conversion into playable music data. *Control and Cybernetics*, 25(2):353–367.

Homenda, W. (2002). Granular computing as an abstraction of data aggregation - a view on optical music recognition. *Archives of Control Sciences*, 12(4):433–455.

Homenda, W. (2005). Optical music recognition: the case study of pattern recognition. In *in: Computer Recognition Systems, Kurzynski et al (Eds.)*, pages 835–842. Springer Verlag.

Homenda, W. and Luckner, M. (2004). Automatic recognition of music notation using neural networks. In *in: Proc. of the International Conference On Artificial Intelligence and Systems, Divnomorskoye, Russia, September 3-10*, pages 74–80, Moscow. Physmathlit.

Homenda, W. and Mossakowski, K. (2004). Music symbol recognition: Neural networks vs. statistical methods. In *in: EUROFUSE Workshop On Data And Knowledge Engineering, Warsaw, Poland, September 22 - 25*, pages 265–271, Warsaw. Physmathlit.

Krolick, B. (1998). *How to Read Braille Music*. Opus Technologies.

McPherson, J. R. (2002). Introducing feedback into an optical music recognition system. In *in: Third Internat. Conf. on Music Information Retrieval, Paris, France*.

Moniuszko, T. (2006). Design and implementation of music processing computer program for blind people (in polish). Master's thesis, Warsaw University of Technology.

Pedrycz, W. (2001). Granular computing: An introduction. In *in: Joint 9th IFSA World Congress and 20th NAFIPS International Conference, Vancouver*, pages 1349–1354.

Taube, H. (1993). Stella: Persistent score representation and score editing in common music. *Computer Music Journal*, 17(3):38–50.

MULTICRITERIA DECISION MAKING IN BALANCED MODEL OF FUZZY SETS

Wladyslaw Homenda
Faculty of Mathematics and Information Science
Warsaw University of Technology, pl. Politechniki 1, 00-661 Warsaw, Poland
homenda@mini.pw.edu.pl

Keywords: Decision making, fuzzy sets, balanced fuzzy sets, negative information.

Abstract: In the paper aspects of negative information and of information symmetry in context of uncertain information processing is considered. Both aspects are presented in frames of fuzzy sets theory involved in data aggregation and decision making process. Asymmetry of classical fuzziness and its orientation to positive information are pointed out. The direct dependence of symmetry of uncertain information on negative information maintenance is indicated. The symmetrical, so called balanced, extension of classical fuzzy sets integrating positive and negative information an paralleling positiveness/negativeness with symmetry of fuzziness is presented. Balanced counterparts of classical fuzzy connectives are introduced.

1 INTRODUCTION

In the paper a discussion on aspects of negative information and information symmetry is presented. The discussion is based on an observation of asymmetry of operators in classical theories of uncertain information and in theories with focus turned on fuzzy sets. Preliminary discussion on asymmetry of classical fuzzy sets is presented in Section 2. It shows an inclination of classical fuzzy connectives to positive information and incompatibility with negative information. Issues related to negative information are outlined in Section 3. Assumed symmetry of negative and positive information integrates both types of information. The integrated approach to parallelism of negativeness/positiveness and symmetry of information is introduced in Section 4. The integration of negativeness and symmetry is inherently drawn in an idea of so called balanced extension of fuzzy sets. This idea was introduced in (Homenda, 2001) and then discussed in several papers (Homenda, 2004; Homenda, 2003; Homenda and Pedrycz, 2002). Finally, an application of the classical approach to uncertainty versus its balanced model is compared in the example 4.1. The example outlines importance of negative information in decision making process in real environment.

2 ASYMMETRY OF FUZZY SETS

Fuzzy set theory is often used to partition a universe into two subsets if partition criteria are not crisp. If partition criterion is uncertain, definition of subsets as fuzzy sets over the universe is a natural way to model uncertainty. However, neither crisp, nor fuzzy modelling avoids problems with law of excluded middle.

Partitioning the universe into two complementary sets suggests comparable significance of both sets unless additional principle is given. In such partitioning elements of the universe can be classified as *true* and *false*, *like* and *dislike*, *good* and *bad*, etc. without any emotional evaluation of these terms. We will simply talk about *positive* and *negative* information, again - without emotional evaluation of both terms. Using several criteria in partitioning we may classify elements of the universe with regard to every criterion separately. Having a number of pairs of complementary sets it is necessary to aggregate these results in order to get final two sets separation of the universe. Using classical aggregators we can choose between *all good criteria* or *one good criterion*. The first one, where the elements classified as *good* one must have all criteria *good*, is implemented by conjunction. The second one, where the element classified as *good* can have only one *good* criterion, is implemented by dis-

junction. Both aggregation connectives, conjunction and disjunction, raise clear asymmetry under complement. If elements of one set are classified as having all criteria *good*, elements of the complementary set must have at least one criterion *bad* instead of expected the same condition of all criteria *bad*. Keeping the same condition (either *all criteria*, or *at least one criterion*) in definition of both sets raises troubles with law of excluded middle mentioned above. Following this way of thinking we need other connectives that will balance aggregation of decisions based on singular criterion. The above discussion leads to the conclusion that classical fuzzy set theory is asymmetrical with regard to processing opposite values of given attributes.

2.1 Symmetrization of the Scale

A classical fuzzy set A in the universe X can be defined in terms of its membership function $\mu : X \rightarrow [0,1]$, where the value 0 means exclusion of the element from the set while the values greater than 0 express the grade of inclusion of the element into the set. However, membership function does not define a grade of exclusion, the grade of negative information. Therefore, fuzzy sets theory distinguishes grades of inclusion and reserves only one value - 0 - for exclusion. This raises asymmetry of this interpretation.

Membership function defines fuzzy connectives: union, intersection and complement. The definitions are expressed by max, min and complement to 1, i.e. $d(x,y) = max\{x,y\}$, $c(x,y) = min\{x,y\}$ and $n(x) = 1-x$. Classical connectives are asymmetrical. Union gets its value from the greater argument, despite of the values of both arguments. Similarly, intersection gets its value from the smallers argument only.

We can split values of a given criterion in the spirit of *good* and *bad* allocating the values of the interval $[0,0.5)$ as pieces of *negative* information relevant to *bad* values and the values of the interval $(0.5.1]$ as pieces of *positive information* relevant to *good* values. The value 0.5, the center of the unit interval $[0,1]$, is a numerical representation of the state of no negative/positive information. Being compatible with common meaning of membership function let us assume that the greater the value of positive information, the stronger the *good* value of the criterion. By symmetry, the smaller the value of negative information, the stronger the *bad* value of the criterion.

This interpretation is well-matched with the common sense of ordering of the negative/positive values. The ordering could be seen as *monotonicity* of negative/positive information mapping: it starts from the left end of the unit interval representing strong negative information, then goes toward middle of the unit interval diminishing strength of negative information, then crosses the middle point of the unit interval and then goes towards the right end of the unit interval increasing strength of positive information.

This interpretation is also well-matched with the common sense of symmetry of the negative/positive values with the symmetry center in the value 0.5. The linear transformation $f(x) = 2x-1$ of the unit interval $[0,1]$ into the symmetrical interval $[-1,1]$ points out the symmetry. In this transformation negative information is mapped to the interval $[-1,0)$, positive information - to the interval $(0,1]$ and the state of no information - to the value 0.

2.2 Connectives Asymmetry

The classical fuzzy connectives stay asymmetrical even with the symmetrical bipolar scale of the interval $[-1,1]$ applied. Both classical fuzzy connectives get their values from the maximal argument (union, maximum) and the minimal argument (intersection, minimum). Classical fuzzy connectives were generalized to triangular norms: maximum is an example of t-conorms, minimum is an example of t-norm, c.f. (Schweizer and Sklar, 1983). Strong t-norms and t-conorms, the special cases of triangular norms, have an interesting property: if both arguments are greater than 0 and smaller than 1, the result of strong t-conorm exceeds the greater argument while the result of strong t-norm is less than smaller argument, c.f. (Klement et al., 2000). This property might be interpreted that union tends to positive information despite of the values of its arguments while intersection tends to negative information despite of the values of its arguments. In other words, symmetrical interpretation of the unipolar scale makes that strong t-norm increases certainty of negative information and decreases certainty of positive information. And vice versa, strong t-conorm decreases certainty of negative information and increases certainty of positive information. This observation emphasizes the asymmetry of fuzzy connectives, c.f. Figure 1.

The problem of asymmetry of fuzzy connectives was discussed in number of papers, e.g. (Detyniecki and Bouchon-Meunier, 2000b; Homenda and Pedrycz, 1991; Homenda and Pedrycz, 2002; Silvert, 1979; Yager, 1988; Yager, 1993; Zhang W. R., 1989). In these papers discussion on asymmetry of fuzzy sets and uncertain information processing was undertaken for different reasons, though common conclusions led to importance of the symmetry problem in fuzziness and uncertainty.

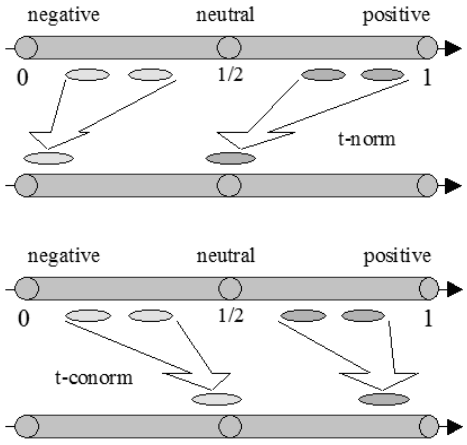


Figure 1: Asymmetry of classical triangular norms.

3 NEGATIVE INFORMATION

The mapping of negative and positive information in the scale of unit interval $[0,1]$ as well as in the symmetrical interval $[-1,1]$ bring incompatibility with connectives, so the question is raised if negative information can be considered as a subject of uncertainty. The question seems justified since negative information is hardly interpretable in classical set theory and classical fuzzy sets theory. However, negative information, as explained in the introductory remarks to this section, play important role in different fields. From psychological studies it is known that human beings convey symmetry in their behavior, c.f. (Grabisch M., 2002). One can be faced with positive (gain, satisfaction, etc.) or negative (loss, dissatisfaction, etc.) quantities, but also with a kind of disinterest (does not matter, not interested in, etc.). For instance, one either likes to listen to the music while reading an interesting novel or does not like to listen to the music then or even music is only a background not affecting him at all. These quantities could be interpreted in context of positive/negative/neutral information. On the other hand, in economy psychological attempt to decision making process with uncertain premises overheads traditional models of customers behavior. The pseudocertainty effect is a concept from prospect theory. It refers to people's tendency to make risk-averse choices if the expected outcome is positive, but risk-seeking choices to avoid negative outcomes. Their choices can be affected by simply reframing the descriptions of the outcomes without changing the actual utility, c.f. (Kahneman and Tversky, 2004). Aggregation of positive and negative premises leads to implementation of a crisp decision. Modelling of such an attempt requires processing of positive/neutral/negative information.

An interesting contribution to positive/negative information maintaining could be found in the theory of intuitionistic fuzzy sets (Atanassov, 1986) and in very similar theory of vague sets (Gau and Buehrer, 1993). Another approach to positive/negative information is discussed in twofold fuzzy sets, c.f. (Dubois and Prade, 1983). In these theories, uncertain information is represented as a pair of positive/negative components numerically described by membership values from the unit interval $[0,1]$. Both components are tied with degree of indeterminacy which stays that sum of membership values of both components cannot exceed the value 1. However, no tool to combine both components is provided in these theories. Since information aggregation leading to non ambiguous result is a clue issue in decision making process, these theories must be supported by information aggregators in such a process.

The very early medical expert system MYCIN, c.f. (Buchanan and Shortliffe, 1984), combine positive and negative information by somewhat ad hock invented aggregation operator. In (Detyniecki, 2000) it was shown that MYCIN aggregation operator is a particular case in a formal study on aggregation of truth and falsity values, c.f. (Detyniecki and Bouchon-Meunier, 2000a) for further discussion on aggregation of positive and negative information.

Having many premisses, usually uncertain, we need to produce nonambiguous information that yields a unique decision. Therefore aggregation of information is crucial in decision making process. The topic of information aggregating has been studied in number of papers. An interesting considerations on information aggregation could be found in - for instance - (Calvo T., 2001; Detyniecki and Bouchon-Meunier, 2000a; Silvert, 1979; Ovchinnikov, 1998; Yager and Rybalov, 1998; Yager and Rybalov, 1996; Zhang W. R., 1989).

4 SYMMETRIZING FUZZINESS

Fuzzy connectives stay asymmetrical with symmetrized scale. The incompatibility of symmetrical interpretation of the scale and asymmetrical behavior of fuzzy connectives suggest incorrectness of scale symmetrization. This discussion leads to the hypothesis that Zadeh's extension of crisp sets to fuzzy sets, c.f. (Zadeh, 1965), relied on dispersion of positive information of the crisp point $\{1\}$ into the interval $(0,1]$. However, negative information of the point $\{0\}$ was still bunched in this point, c.f. Figure 2. This hypothesis can be supported by similarity of balanced

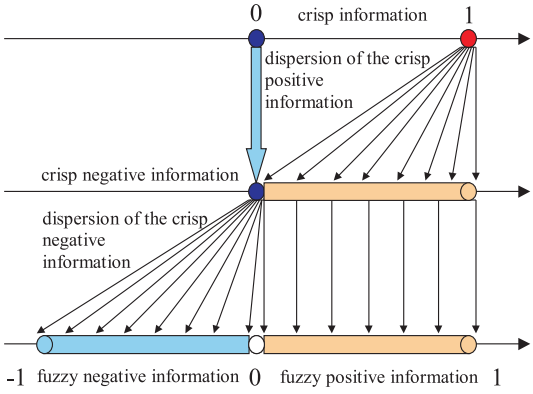


Figure 2: Extension of fuzzy sets to balanced fuzzy sets.

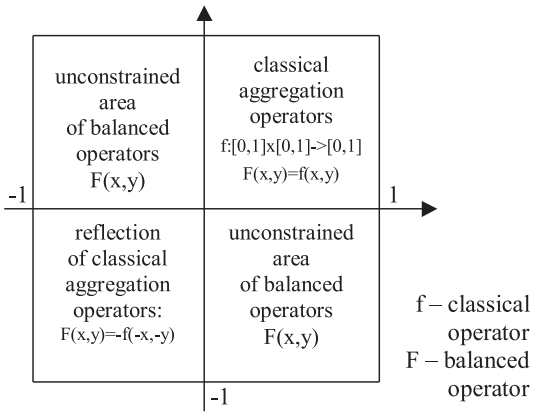


Figure 3: Balanced extension of fuzzy operators.

triangular norms and uninorms and nullnorms - the products of different approaches to fuzzy connectives extension, c.f. (Homenda, 2003)

4.1 Balanced Symmetrization of the Scale

Now, the process of information dispersion is applied again to information concentrated in the point $\{0\}$. This operation extends classical fuzzy sets to balanced fuzzy sets, c.f. (Homenda, 2004; Homenda, 2003). The extension is being done by dispersion of crisp negative information bunched in the point $\{0\}$ into the interval $[-1.0)$ without affecting classical fuzzy sets based on the unit interval $[0,1]$ c.f. Figure 2. Thus, classical fuzzy sets will be immersed in a new space of balanced fuzzy sets. Since both types of information - positive and negative - are assumed to be equally important, it would be reasonable to expect that such an extension will provide a kind of symmetry of positive/negative information.

Concluding, the following symmetry principle can be formulated: *the extension of fuzzy sets to balanced fuzzy sets relies on spreading negative information (information about exclusion) that fit the crisp point* $\{0\}$ *of fuzzy set into the interval* $[-1,0)$. The extension will preserve properties of classical operators for positive information. It will provide the symmetry of positive/negative information with the center of symmetry placed in the point 0, c.f. Figure 2. It is worth to underline that this operation is entirely different than simple linear rescaling of the unit interval $[0,1]$ into the interval $[-1,1]$. The linear function $f(x) = 2x-1$ is replaced by the transformation that is not a function: it allocates the whole interval $[-1,0)$ as a "value" in the point 0.

4.2 Symmetry of Balanced Connectives

Triangular norms generalize the concept of set operations: union and intersection, c.f. (Schweizer and Sklar, 1983). Triangular norms, t-norms and t-conorms, together with negation, the basic fuzzy connectives are the subject of the discussion of connectives symmetrization.

Definition 4.1 *Triangular norms: t-norm t and t-conorm s, are mappings $t : [0,1] \times [0,1] \to [0,1]$ and $s : [0,1] \times [0,1] \to [0,1]$ satisfying the following axioms:*

1. $t(a,t(b,c)) = t(t(a,b),c)$
   $s(a,s(b,c)) = s(s(a,b),c)$ — *associativity*
2. $t(a,b) = t(b,a)$
   $s(a,b) = s(b,a)$ — *commutativity*
3. $t(a,b) \leq t(c,d)$ *if* $a \leq c$ & $b \leq d$
   $s(a,b) \leq s(c,d)$ *if* $a \leq c$ & $b \leq d$ — *monotonicity*
4. $t(1,a) = a$ *for* $a \in [0,1]$
   $s(0,a) = a$ *for* $a \in [0,1]$ — *boundary conditions*

t-norms and t-conorms are dual operations in the sense that for any given t-norm t and given negation operator assumed here to be complement to one, we have the dual t-conorm s defined by the De Morgan formula $s(a,b) = 1-t(1-a,1-b)$. And vice-versa, for any given t-conorm s, we have the dual t-norm t defined by the De Morgan formula $t(a,b) = 1-s(1-a,1-b)$. Duality of triangular norms causes duality of their properties. Note that the max/min is a pair of dual t-norm and t-conorm.

The idea of balanced extension of classical fuzzy connectives must be compatible with the concept of balanced extension of the unipolar scale and with the symmetry principle formulated in Section 4.1. This requirements and symmetry of the balanced fuzzy scale of the interval $[-1,1]$ determines the domain of

symmetrized balanced connectives to be the square $[-1,1]\times[-1,1]$. Preservation of classical fuzzy sets properties requires preservation of properties of classical fuzzy connectives on the unit square $[0,1]\times[0,1]$. Conversely, expected symmetry of positive and negative information puts strict restrictions on balanced extension on the square $[-1,0]\times[-1,0]$. The same factors determine the co-domain of symmetrical fuzzy connectives to the interval $[-1,1]$. This idea of the balanced extension of classical fuzzy connectives is outlined in Figure 3. It is clear that balanced connectives are simple reflection of respective classical connectives on the square $[-1,0]\times[-1,0]$. The remaining parts of the domain of balanced connectives are not explicitly constrained. However, some constrains will be put when other properties of connectives are considered. These properties come from natural extension of the axioms of the triangular norms definition onto the whole domain of balanced operators.

This discussion leads to the definition of balanced negation, balanced t-norms and balanced t-conorms.

Definition 4.2 *of balanced connectives.*
The mapping $N:[-1,1]\to[-1,1]$, $N(x)=-x$ *is the balanced negation.*
The mappings $T:[-1,1]\times[-1,1]\to[-1,1]$ *and* $S:[-1,1]\times[-1,1]\to[-1,1]$ *are balanced t-norm and balanced t-conorm, respectively, assuming that they satisfy the following axioms in the whole domain* $[-1,1]\times[-1,1]$ *unless defined explicitly:*

1., 2., 3.
associativity, commutativity and monotonicity

4. $T(1,a)=a \quad for \quad a\in[0,1]$ *boundary*
   $S(0,a)=a \quad for \quad a\in[0,1]$ *conditions*
5. $T(x,y)=N(T(N(x),N(y)))$
   $S(x,y)=N(S(N(x),N(y)))$ *symmetry*

Conclusion 4.1 *The definitions of balanced t-norm and balanced t-conorm restricted to the unit square* $[0,1]\times[0,1]$ *are equivalent to the classical t-norm and classical t-conorm, respectively.*

Conclusion 4.2 *Balanced t-norm and balanced t-conorm restricted to the square* $[-1,0]\times[-1,0]$ *are isomorphic with the classical t-conorm and classical t-norm, respectively.*

Conclusion 4.3 *Balanced t-norm vanishes on the squares* $[-1,0]\times[0,1]$ *and* $[0,1]\times[-1,0]$.

Example 4.1 *Let us consider the strong t-norm generated by the additive generator* $f(x)=(1-x)/x$ *and the strong t-conorm generated by the additive generator* $f(x)=x/(1-|x|)$, *c.f. (Klement et al., 2000; Schweizer and Sklar, 1983). The formula* $p(x,y)=f^{-1}(f(x)+f(y))$ *defines the respective strong triangular norms. The extension of this t-norm to balanced t-norm comes from the monotonicity and symmetry axioms directly. Alternatively, the additive generator* $f(x)=(1-x)/x$ *of this classical t-norm could be extended to the interval* $[-1,1]$ *with the formula* $f(x)=(1-|x|)/x$. *Of course, this function is undefined in the point* 0. *The formula* $p(x,y)=f^{-1}(f(x)+f(y))$ *defines the balanced t-norm for both arguments being nonnegative or nonpositive.*

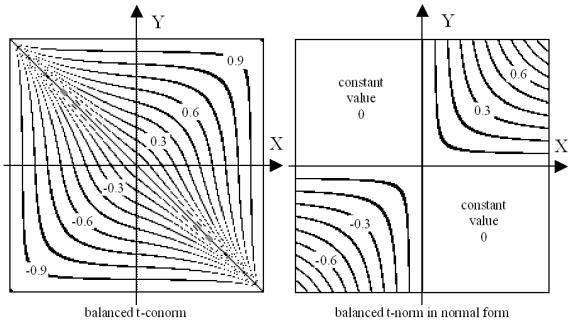


Figure 4: Balanced t-conorm and balanced t-norm.

The balanced counterpart of the strong t-conorm generated by the additive generator $f(x)=x/(1-|x|)$ *is determined in the squares* $[0,1]\times[0,1]$ *and* $[-1,0]\times[-1,0]$. *The values in remaining parts of the domain are unconstrained besides that they must satisfy axioms of the definition. In this case we can extend the additive generator to the whole interval* $[-1,1]$ *assuming that in points* -1 *and* 1 *it gets the values* $-\infty$ *and* ∞, *respectively. The balanced t-conorm could be defined by the formula* $S(x,y)=f^{-1}(f(x)+f(y))$ *in its whole domain* $[-1,1]\times[-1,1]$ *except the points* $(-1,1)$ *and* $(1,-1)$, *where this balanced t-conorm is undefined, c.f. (Homenda, 2003; Klement et al., 2000; Schweizer and Sklar, 1983) for details. Contour graphs of these two balanced norms are presented in Figure 4.*

5 BALANCED FUZZY SETS IN DECISION MAKING

Let us consider simple decision making process in real economical environment, i.e. with uncertain premises. Assume that in the first set of six premises five have the numerical value 0.6 and the last one has the value 0.4. In the second set of six premises the one has numerical value 0.6 and five other have numerical values 0.4. Classical fuzzy connectives employed as aggregators of premises do not distinguish between these two sets. Max and min operators produce the same values for both sets: 0.6 and 0.4, respectively. Employing the strong t-conorm based on

the additive operator $f(x) = x/(1-x)$ we get the values 0.89 and 0.83 for both sets, respectively. In the case of dual t-norm we get the respective values equal to 0.17 and 0.11. The linear mapping $f(x) = 2x-1$ to the interval $[-1,1]$ gives the transformed values of premises equal to 0.2 and -0.2 instead of 0.6 and 0.4. Classical triangular norms produce the following the respective values: 0.78, 0.66 and -0.66, -0.78. Therefore, besides small quantitative differences between aggregated numerical results no qualitative indication is given with regard to the decision. Employing balanced modelling based on the additive operator $f(x) = x/(1-|x|)$ balanced t-conorm produces the values 0.50 and -0.50 for both sets of premises, respectively. Dual balanced t-norm produces the values 0.06 and -0.06, respectively. In the case of balanced modelling clear indication is given with regard to the decision.

6 CONCLUSIONS

The balanced extension of fuzzy sets discussed in this paper is a contribution to the discussion on subjects of negative information and symmetry of negative/positive types of information. These aspects of information processing, though controversial in classical and traditional fields of information processing, become useful and necessary in some important areas of research and practice, as indicated in Section 3 and have been studied in number of papers.

Negative and positive types information play important role in information aggregation. Multicriteria decision making process could be seen as a kind of information aggregation leading to a synthetical result applicable in unique choice between given options. The synthesis must consider pros and contras a decision, must consider positive and negative premisses of the decision. The concept of balanced fuzzy sets deals with positiveness and negativeness assuming symmetry of both types of information.

In (Homenda and Pedrycz, 2005) the concept of negativeness and symmetry was applied in construction of the balanced computing unit, a variation of fuzzy neuron. The concept of balanced computing unit involves a generalization of balanced t-norms, so called t-norms in weak form. The balanced computing unit based on weak form of fuzzy connectives may exemplify decision making process with positive and negative premises.

REFERENCES

Atanassov, K. T. (1986). Intuitionistic fuzzy sets. *Fuzzy Sets and Systems*, 20:87–96.

Buchanan, B. and Shortliffe, E. (1984). *Ruled Based Expert Systems, The MYCIN Experiment of the Stanford Heuristic Programming Project*. Addison-Wesley, Reading, MA.

Calvo T., e. a. (2001). The functional equations of frank and alsina for uninorms and nullnorms. *Fuzzy Sets and Systems*, 120:15–24.

Detyniecki, M. (2000). *Mathematical Aggregation Operators and their Application to Video Querying*. PhD thesis, l'Universite Paris VI.

Detyniecki, M. and Bouchon-Meunier, B. (2000a). Aggregating truth and falsity values. In *Proc. of the Int. Conf. on Information Fusion - FUSION'2000, Paris, France, July 2000*, pages 18–24.

Detyniecki, M. and Bouchon-Meunier, B. (2000b). Building an aggregation operator with a balance. In *Proc. of the Int. Conf. on Information Processing and Management of Uncertainty in Knowledge-Based Systems, Madrid, Spain, July 2000*, pages 686–692.

Dubois, D. and Prade, H. (1983). Twofold fuzzy sets: an approach to the representation of sets with fuzzy boundaries based on possibility and necessity measures. *Journal of Fuzzy Mathematics*, 3(4):53–76.

Gau, W. L. and Buehrer, D. J. (1993). Vague sets. *IEEE Transactions on Systems, Man, and Cybernetics*, 23:610–614.

Grabisch M., e. a. (2002). On symmetric pseudo-additions and pseudo-multiplications: Is it possible to built rings on [-1,+1]? In *The 9th Int. Conf. on Information Processing and Management of Uncertainty in Knowledge-Based Systems, IPMU 2002, 1-5 July, France*.

Homenda, W. (2001). Balanced fuzzy sets. Preprint, Faculty of Mathematics and Information Science, Warsaw University of Technology.

Homenda, W. (2003). Triangular norms, uni- and nullnorms, balanced norms: the cases of the hierarchy of iterative operators. In *Proc. of the 24th Linz Seminar on Fuzzy Set Theory, Linz, Feb. 4-8*, pages 27–35.

Homenda, W. (2004). *Balanced norms: from triangular norms towards iterative operators*, pages 251–262. in: Soft Computing, Foundations and theoretical aspects. Academic Publishing House EXIT, Warsaw.

Homenda, W. and Pedrycz, W. (1991). Processing of uncertain information in linear space of fuzzy sets. *Fuzzy Sets and Systems*, 44:187–198.

Homenda, W. and Pedrycz, W. (2002). Symmetrization of fuzzy operators: notes on data aggregation. In *Proc. of the Int. Conf. on Fuzzy Systems and Knowledge Discovery, Singapore, Nov. 18-22*.

Homenda, W. and Pedrycz, W. (2005). Balanced fuzzy computing unit. *Int. Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 13(2):117–138.

Kahneman, D. and Tversky, A. (2004). *Prospect Theory: An Analysis of Decision under Risk*, volume XVLII of *in: Soft Computing, Foundations and theoretical aspects*, pages 263–291. Econometrica.

Klement, E. P., Mesiar, R., and Pap, E. (2000). *Triangular norms*. Kluwer Academic Pub., Dordrecht.

Ovchinnikov, S. (1998). *On Robust Aggregation Procedures*, pages 3–10. Econometrica.

Schweizer, B. and Sklar, A. (1983). *Probabilistic Metric Spaces*. North Holland, New York.

Silvert, W. (1979). Symmetric summation: A class of operations on fuzzy sets. *IEEE Trans. System, Man, Cybernetics*, 9:659–667.

Yager, R. R. (1988). On ordered weighted averaging aggregation operators. *IEEE Trans. System, Man, Cybernetics*, 18:183–190.

Yager, R. R. (1993). Families of owa operators. *Fuzzy Sets and Systems*, 59:125–148.

Yager, R. R. and Rybalov, A. (1996). Uninorm aggregation operators. *Fuzzy Sets and Systems*, 80:111–120.

Yager, R. R. and Rybalov, A. (1998). Full reinforcement operators in aggregation techniques. *IEEE Transactions on Systems, Man and Cybernetics*, 28:757–769.

Zadeh, L. A. (1965). Fuzzy sets. *Inform. And Control*, 8:338–353.

Zhang W. R., e. a. (1989). Pool2, a generic system for cognitive map development and decision analysis. *IEEE Tran. on Systems, Man and Cybernetics*, 19:31–39.

NONLINEAR PROGRAMMING IN APPROXIMATE DYNAMIC PROGRAMMING

Bang-bang Solutions, Stock-management and Unsmooth Penalties

Olivier Teytaud and Sylvain Gelly
TAO (Inria, Univ. Paris-Sud, UMR CNRS-8623)
olivier.teytaud@inria.fr, gelly@lri.fr

Keywords: Evolutionary computation and control, Optimization algorithms.

Abstract: Many stochastic dynamic programming tasks in continuous action-spaces are tackled through discretization. We here avoid discretization; then, approximate dynamic programming (ADP) involves (i) many learning tasks, performed here by Support Vector Machines, for Bellman-function-regression (ii) many non-linear-optimization tasks for action-selection, for which we compare many algorithms. We include discretizations of the domain as particular non-linear-programming-tools in our experiments, so that by the way we compare optimization approaches and discretization methods. We conclude that robustness is strongly required in the non-linear-optimizations in ADP, and experimental results show that (i) discretization is sometimes inefficient, but some specific discretization is very efficient for "bang-bang" problems (ii) simple evolutionary tools outperform quasi-random in a stable manner (iii) gradient-based techniques are much less stable (iv) for most high-dimensional "less unsmooth" problems Covariance-Matrix-Adaptation is first ranked.

1 NON-LINEAR OPTIMIZATION IN STOCHASTIC DYNAMIC PROGRAMMING (SDP)

Some of the most traditional fields of stochastic dynamic programming, e.g. energy stock-management, which have a strong economic impact, have not been studied thoroughly in the reinforcement learning or approximate-dynamic-programming (ADP) community. This is damageable to reinforcement learning as it has been pointed out that there are not yet many industrial realizations of reinforcement learning. Energy stock-management leads to continuous problems that are usually handled by traditional linear approaches in which (i) convex value-functions are approximated by linear cuts (leading to piecewise linear approximations (PWLA)) (ii) decisions are solutions of a linear-problem. However, this approach does not work in large dimension, due to the curse of dimensionality which strongly affects PWLA. These problems should be handled by other learning tools. However, in this case, the action-selection, minimizing the expected cost-to-go, can't be anymore done using linear-programming, as the Bellman function is no more a convex PWLA.

The action selection is therefore a nonlinear programming problem. There are not a lot of works dealing with continuous actions, and they often do not study the non-linear optimization step involved in action selection. In this paper, we focus on this part: we compare many non-linear optimization-tools, and we also compare these tools to discretization techniques to quantify the importance of the action-selection step.

We here roughly introduce stochastic dynamic programming. The interested reader is referred to (Bertsekas and Tsitsiklis, 1996) for more details.

Consider a dynamical system that stochastically evolves in time depending upon your decisions. Assume that time is discrete and has finitely many time steps. Assume that the total cost of your decisions is the sum of instantaneous costs. Precisely:

$$\begin{aligned} cost &= c_1 + c_2 + \cdots + c_T \\ c_i &= c(x_i, d_i), \quad x_i = f(x_{i-1}, d_{i-1}, \omega_i) \\ d_{i-1} &= strategy(x_{i-1}, \omega_i) \end{aligned}$$

where x_i is the state at time step i, the ω_i are a random process, *cost* is to be minimized, and *strategy* is the decision function that has to be optimized. We are interested in a control problem: the element to be optimized is a function.

Stochastic dynamic programming, a tool to solve this control problem, is based on Bellman's optimality

principle that can be informally stated as follows:

"Take the decision at time step t such that the sum "cost at time step t due to your decision" plus "expected cost from time step $t+1$ to ∞" is minimal."

Bellman's optimality principle states that this strategy is optimal. Unfortunately, it can only be applied if the expected cost from time step $t+1$ to ∞ can be guessed, depending on the current state of the system and the decision. Bellman's optimality principle reduces the control problem to the computation of this function. If x_t can be computed from x_{t-1} and d_{t-1} (i.e., if f is known) then the control problem is reduced to the computation of a function

$$V(t,x_t) = E[c(x_t,d_t) + c(x_{t+1},d_{t+1}) + \cdots + c(x_T,d_T)]$$

Note that this function depends on the strategy (we omit for short dependencies on the random process). We consider this expectation for any optimal strategy (even if many strategies are optimal, V is uniquely determined as it is the same for any optimal strategy).

Stochastic dynamic programming is the computation of V backwards in time, thanks to the following equation:

$$\begin{aligned} V(t,x_t) &= \inf_{d_t} c(x_t,d_t) + EV(t+1,x_{t+1}) \\ &\text{or equivalently} \\ V(t,x_t) &= \inf_{d_t} c(x_t,d_t) + EV(t+1,f(x_t,d_t)) \quad (1) \end{aligned}$$

For each t, $V(t,x_t)$ is computed for many values of x_t, and then a learning algorithm (here by support vector machines) is applied for building $x \mapsto V(t,x)$ from these examples. Thanks to Bellman's optimality principle, the computation of V is sufficient to define an optimal strategy. This is a well known, robust solution, applied in many areas including power supply management. A general introduction, including learning, is (Bertsekas, 1995; Bertsekas and Tsitsiklis, 1996). Combined with learning, it can lead to positive results in spite of large dimensions. Many developments, including RTDP and the field of reinforcement learning, can be found in (Sutton and Barto, 1998).

Equation 1 is used many many times during a run of dynamic programming. For T time steps, if N points are required for efficiently approximating each V_t, then there are $T \times N$ optimizations. Furthermore, the derivative of the function to optimize is not always available, due to the fact that complex simulators are sometimes involved in the transition f. Convexity sometimes holds, but sometimes not. Binary variables are sometimes involved, e.g. in power plants management. This suggests that evolutionary algorithms are a possible tool.

1.1 Robustness in Non-linear Optimization

Robustness is one of the main issue in non-linear optimization and has various meanings.

1. A first meaning is the following: robust optimization is the search of x such that in the neighborhood of x the fitness is good, and not only at x. In particular, (DeJong, 1992) has introduced the idea that evolutionary algorithms are not function-optimizers, but rather tools for finding wide areas of good fitness.

2. A second meaning is that robust optimization is the avoidance of local minima. It is known that iterative deterministic methods are often more subject to local minima than evolutionary methods; however, various forms of restarts (relaunch the optimization from a different initial point) can also be efficient for avoiding local minima.

3. A third possible meaning is the robustness with respect to fitness noise. Various models of noise and conclusions can be found in (Jin and Branke, 2005; Sendhoff et al., 2004; Tsutsui, 1999; Fitzpatrick and Grefenstette, 1988; Beyer et al., 2004).

4. A fourth possible meaning is the robustness with respect to unsmooth fitness functions, even in cases in which there's no local minima. Evolutionary algorithms are usually rank-based (the next iterate point depends only on the fitnesses rank of previously visited points), therefore do not depend on increasing transformations of the fitness-function. It is known that they have optimality properties w.r.t this kind of transformations (Gelly et al., 2006). For example, $\sqrt{||x||}$ (or some C^∞ functions close to this one) lead to a very bad behavior of standard Newton-based methods like BFGS (Broyden., 1970; Fletcher, 1970; Goldfarb, 1970; Shanno, 1970) whereas a rank-based evolutionary algorithm behaves the same for $||x||^2$ and $\sqrt{||x||}$.

5. The fifth possible meaning is the robustness with respect to the non-deterministic choices made by the algorithm. Even algorithms that are considered as deterministic often have a random part[1]: the choice of the initial point. Population-based methods are more robust in this sense, even if they use more randomness for the initial step (full random initial population compared to only one initial point): a bad initialization which would lead to a disaster is much more unlikely.

The first sense of robustness given above, i.e. avoiding too narrow areas of good fitness, fully applies here. Consider for example a robot navigating

[1]Or, if not random, a deterministic but arbitrary part, such as the initial point or the initial step-size.

in an environment in order to find a target. The robot has to avoid obstacles. The strict optimization of the cost-to-go leads to choices just tangent to obstacles. As at each step the learning is far from perfect, then being tangent to obstacles leads to hit the obstacles in 50 % of cases. We see that some local averaging of the fitness is suitable.

The second sense, robustness in front of non-convexity, of course also holds here. Convex and non-convex problems both exist. The law of increasing marginal costs implies the convexity of many stock management problems, but approximations of V are usually not convex, even if V is theoretically convex. Almost all problems of robotics are not convex.

The third sense, fitness (or gradient) noise, also applies. The fitness functions are based on learning from finitely many examples. Furthermore, the gradient, when it can be computed, can be pointless even if the learning is somewhat successfull; even if $\hat{f}$ approximates f in the sense that $||f-\hat{f}||_p$ is small, $\nabla\hat{f}$ can be very far from ∇f.

The fourth sense is also important. Strongly discontinuous fitnesses can exist: obstacle avoidance is a binary reward, as well as target reaching. Also, a production-unit can be switched on or not, depending on the difference between demand and stock-management, and that leads to large binary-costs.

The fifth sense is perhaps the most important. SDP can lead to thousands of optimizations, similar to each other. Being able of solving very precisely 95 % of families of optimization problems is not the goal; here it's better to solve 95 % of any family of optimization problems, possibly in a suboptimal manner. We do think that this requirement is a main explanation of results below.

Many papers have been devoted to ADP, but comparisons are usually far from being extensive. Many papers present an application of one algorithm to one problem, but do not compare two techniques. Problems are often adapted to the algorithm, and therefore comparing results is difficult. Also, the optimization part is often neglected; sometimes not discussed, and sometimes simplified to a discretization.

In this paper, we compare experimentally many non-linear optimization-tools. The list of methods used in the comparison is given in 2. Experiments are presented in section 3. Section 4 concludes.

2 ALGORITHMS USED IN THE COMPARISON

We include in the comparison standard tools from mathematical programming, but also evolutionary algorithms and some discretization techniques. Evolutionary algorithms can work in continuous domains (Bäck et al., 1991; Bäck et al., 1993; Beyer, 2001); moreover, they are compatible with mixed-integer programming (e.g. (Bäck and Schütz, 1995)). However, as there are not so many algorithms that could naturally work on mixed-integer problems and in order to have a clear comparison with existing methods, we restrict our attention to the continuous framework. We can then easily compare the method with tools from derivative free optimization (Conn et al., 1997), and limited-BFGS with finite differences (Zhu et al., 1994; Byrd et al., 1995). We also considered some very naive algorithms that are possibly interesting thanks to the particular requirement of robustness within a moderate number of iterates: random search and some quasi-random improvements. The discretization techniques are techniques that test a predefined set of actions, and choose the best one. As detailed below, we will use dispersion-based samplings or discrepancy-based samplings.

We now provide details about the methods integrated in the experiments. For the sake of neutrality and objectivity, none of these source codes has been implemented for this work: they are all existing codes that have been integrated to our platform, except the baseline algorithms.

- random search: randomly draw N points in the domain of the decisions ; compute their fitness ; consider the minimum fitness.
- quasi-random search: idem, with low discrepancy sequences instead of random sequences (Niederreiter, 1992). Low discrepancy sequences are a wide area of research (Niederreiter, 1992; Owen, 2003), with clear improvements on Monte-Carlo methods, in particular for integration but also for learning (Cervellera and Muselli, 2003), optimization (Niederreiter, 1992; Auger et al., 2005), path planning (Tuffin, 1996). Many recent works are concentrated on high dimension (Sloan and Woźniakowski, 1998; Wasilkowski and Wozniakowski, 1997), with in particular successes when the "true" dimensionality of the underlying distribution or domain is smaller than the apparent one (Hickernell, 1998), or with scrambling-techniques (L'Ecuyer and Lemieux, 2002).
- Low-dispersion optimization is similar, but uses low-dispersion sequences (Niederreiter, 1992; Lindemann and LaValle, 2003; LaValle et al., 2004) instead of random i.i.d sequences ; low-dispersion is related to low-discrepancy, but easier to optimize. A dispersion-criterion is

$$Dispersion(P) = \sup_{x\in D} \inf_{p\in P} d(x,p) \qquad (2)$$

where d is the euclidean distance. It is related to the following (easier to optimize) criterion (to be maximized and not minimized):

$$Dispersion_2(P) = \inf_{(x_1,x_2)\in D^2} d(x_1,x_2) \quad (3)$$

we use eq. 3 in the sequel of this paper. We optimize dispersion in a greedy manner: each point x_n is optimal for the dispersion of $x_1,\ldots,x_n$ conditionally to $x_1,\ldots,x_{n-1}$; i.e. $x_1 = (0.5,0.5,\ldots,0.5)$, x_2 is such that $Dispersion_2(\{x_1,x_2\})$ is maximal, and x_n is such that $Dispersion_2(\{x_1,\ldots,x_{n-1},x_n\})$ is minimal. This sequence has the advantage of being much faster to compute than the non-greedy one, and that one does not need a priori knowledge of the number of points. Of course, it is not optimal for eq. 3 or eq. 2.

- Equation 3 pushes points on the frontier, what is not the case in equation 2 ; therefore, we also considered low-dispersion sequences "far-from-frontier", where equation 3 is replaced by:

$$Dispersion_3(P) = \inf_{(x_1,x_2)\in D^2} d(x_1,\{x_2\}\cup D') \quad (4)$$

As for $Dispersion_2$, we indeed used the greedy and incremental counterpart of eq. 4.

- CMA-ES (EO and openBeagle implementation): an evolution strategy with adaptive covariance matrix (Hansen and Ostermeier, 1996; Keijzer et al., 2001; Gagné, 2005).
- The Hooke & Jeeves (HJ) algorithm (Hooke and Jeeves, 1961; Kaupe, 1963; Wright, 1995), available at `http://www.ici.ro/camo/unconstr/hooke.htm`: a geometric local method implemented in C by M.G. Johnson.
- a genetic algorithm (GA), from the *sgLibrary* (`http://opendp.sourceforge.net`). It implements a very simple genetic algorithm where the mutation is an isotropic Gaussian of standard deviation $\frac{\sigma}{\sqrt[d]{n}}$ with n the number of individuals in the population and d the dimension of space. The crossover between two individuals x and y gives birth to two individuals $\frac{1}{3}x+\frac{2}{3}y$ and $\frac{2}{3}x+\frac{1}{3}y$. Let $\lambda_1,\lambda_2,\lambda_3,\lambda_4$ be such that $\lambda_1+\lambda_2+\lambda_3+\lambda_4 = 1$; we define S_1 the set of the $\lambda_1.n$ best individuals, S_2 the $\lambda_2.n$ best individuals among the others. At each generation, the new offspring is (i) a copy of S_1 (ii) $n\lambda_2$ cross-overs between individuals from S_1 and individuals from S_2 (iii) $n\lambda_3$ mutated copies of individuals from S_1 (iv) $n\lambda_4$ individuals randomly drawn uniformly in the domain. The parameters are $\sigma = 0.08, \lambda_1 = 1/10, \lambda_2 = 2/10, \lambda_3 = 3/10, \lambda_4 = 4/10$; the population size is the square-root of the number of fitness-evaluations allowed. These parameters are standard ones from the library. We also use a "no memory" (GANM) version, that provides as solution the best point in the final offspring, instead of the best visited point. This is made in order to avoid choosing a point from a narrow area of good fitness.
- limited-BFGS with finite differences, thanks to the LBFGSB library (Zhu et al., 1994; Byrd et al., 1995). Roughly speaking, LBFGS uses an approximated Hessian in order to approximate Newton-steps without the huge computational and space cost associated to the use of a full Hessian.

In our experiments with restart, any optimization that stops due to machine precision is restarted from a new random (independent, uniform) point.

For algorithms based on an initial population, the initial population is chosen randomly (uniformly, independently) in the domain. For algorithms based on an initial point, the initial point is the middle of the domain. For algorithms requiring step sizes, the step size is the distance from the middle to the frontier of the domain (for each direction). Other parameters were chosen by the authors with equal work for each method on a separate benchmark, and then plugged in our dynamic programming tool. The detailed parametrization is available in `http://opendp.sourceforge.net`, with the command-line generating tables of results.

Some other algorithms have been tested and rejected due to their huge computational cost: the DFO-algorithm from Coin (Conn et al., 1997),`http://www.coin-or.org/`; Cma-ES from Beagle (Hansen and Ostermeier, 1996; Gagné, 2005) is similar to Cma-ES from EO(Keijzer et al., 2001) and has also been removed.

3 EXPERIMENTS

3.1 Experimental Settings

The characteristics of the problems are summarized in table 1; problems are scalable and experiments are performed with dimension (i) the baseline dimension in table 1 (ii) twice this dimensionality (iii) three times (iv) four times. Both the state space dimension and the action space are multiplied. Results are presented in tables below. The detailed experimental setup is as follows: the learning of the function value is performed by SVM with Laplacian-kernel (SVMTorch,(Collobert and Bengio, 2001)), with hyper-

parameters heuristically chosen; each optimizer is allowed to use a given number of points (specified in tables of results); 300 points for learning are sampled in a quasi-random manner for each time step, non-linear optimizers are limited to 100 function-evaluations. Each result is averaged among 66 runs. We can summarize results below as follows. Experiments are performed with:

- 2 algorithms for gradient-based methods (LBFGS and LBFGS with restart),
- 3 algorithms for evolutionary algorithms (EO-CMA, GeneticAlgorithm, GeneticAlgorithm-NoMemory),
- 4 algorithms for best-of-a-predefined sample (Low-Dispersion, Low-Dispersion "fff", Random, Quasi-Random),
- 2 algorithms for pattern-search methods (Hooke&Jeeves, Hooke&Jeeves with restart)

3.2 Results

Results varies from one benchmark to another. We have a wide variety of benchmarks, and no clear superiority of one algorithm onto others arises. E.g., CMA is the best algorithm in some cases, and the worst one in some others. One can consider that it would be better to have a clear superiority of one and only one algorithm, and therefore a clear conclusion. Yet, it is better to have plenty of benchmarks, and as a by-product of our experiments, we claim that conclusions extracted from one or two benchmarks, as done in some papers, are unstable, in particular when the benchmark has been adapted to the question under study. The significance of each comparison (for one particular benchmark) can be quantified and in most cases we have sufficiently many experiments to make results significant. But, this significance is for each benchmark independently; in spite of the fact that we have chosen a large set of benchmarks, coming from robotics or industry, we can not conclude that the results could be extended to other benchmarks. However, some (relatively) stable conclusions are:

- For best-of-a-predefined-set-of-points:
  - Quasi-random search is better than random search in 17/20 experiments with very good overall significance and close to random in the 3 remaining experiments.
  - But low-dispersion, that is biased in the sense that it "fills" the frontier, is better in 10 on 20 benchmarks only; this is problem-dependent, in the sense that in the "away" or "arm" problem, involving nearly bang-bang solutions (i.e. best actions are often close to the boundary for each action-variable) the Low-dispersion-approach is often the best. LD is the best with strong significance for many *-problems (in which bang-bang solutions are reasonnable).
  - And low-dispersion-fff, that is less biased, outperforms random for 14 on 20 experiments (but is far less impressive for bang-bang-problems).
- For order-2 techniques[2]: LBFGSB outperforms quasi-random-optimization for 9/20 experiments; Restart-LBFGSB outperforms quasi-random optimization for 10/20 experiments. We suggest that this is due to (i) the limited number of points (ii) the non-convex nature of our problems (iii) the cost of estimating a gradient by finite-differences that are not in favor of such a method. Only comparison-based tools were efficient. CMA is a particular tool in the sense that it estimates a covariance (which is directly related to the Hessian), but without computing gradients; a drawback is that CMA is much more expensive (much more computation-time per iterate) than other methods (except BFGS sometimes). However it is sometimes very efficient, as being a good compromise between a precise information (the covariance related to the Hessian) and fast gathering of information (no gradient computation). In particular, CMA was the best algorithm for all stock-management problems (involving precise choices of actions) as soon as the dimension is ≥ 8, with in most cases strong statistical significance.
- The pattern-search method (the Hooke&Jeeves algorithm with Restart) outperforms quasi-random for 10 experiments on 20.
- For the evolutionary-algorithms:
  - EoCMA outperforms Quasi-Random in 5/20 experiments. These 5 experiments are all stock-management in high-dimension, and are often very significant.
  - GeneticAlgorithm outperforms Quasi-Random in 14/20 experiments and Random in 17/20 experiments (with significance in most cases). This algorithm is probably the most stable one in our experiments. GeneticAlgorithmNoMemory outperforms Quasi-Random in 14/20 experiments and Random in 15/20 experiments.

Due to length-constraints, the detailed results, for each method and with confidence intervals,

[2] We include CMA in order-2 techniques in the sense that it uses a covariance matrix which is strongly related to the Hessian.

are reported to http://www.lri.fr/~teytaud/sefordplong.pdf. We summarize the results in table 2.

4 CONCLUSION

We presented an experimental comparison of non linear optimization algorithms in the context of ADP. The comparison involves evolutionary algorithms, (quasi-)random search, discretizations, and pattern-search-optimization. ADP has strong robustness requirements, thus the use of evolutionary algorithms, known for their robustness properties, is relevant. These experiments are made in a neutral way; we did not work more on a particular algorithm than another. Of course, perhaps some algorithms require more work to become efficient on the problem. The reader can download our source code, modify the conditions, check the parametrization, and experiment himself. Therefore, our source code is freely available at http://opendp.sourceforge.net for further experiments. A Pascal-NoE challenge (www.pascal-network.org/Challenges/) will be launched soon so that anyone can propose his own algorithms.

Our main claims are:

- **High-dimensional stock-management**. CMA-ES is an efficient evolution-strategy when dimension increases and for "less-unsmooth" problems. It is less robust than the GA, but appears as a very good compromise for the important case of high-dimensional stock-management problems. We do believe that CMA-ES, which is very famous in evolution strategies, is indeed a very good candidate for non-linear optimization as involved in high-dimensional-stock-management where there is enough smoothness for covariance-matrix-adaptation. LBFGS is not satisfactory: in ADP, convexity or derivability are not reliable assumptions, as explained in section 1.1, even if the law of increasing marginal cost applies. Experiments have been performed with dimension ranging from 4 to 16, without heuristic dimension reduction or problem-rewriting in smaller dimension, and results are statistically clearly significant. However, we point out that CMA-ES has a huge computational cost. The algorithms are compared above in the case of a given number of calls to the fitness; this is only a good criterion when the computational cost is mainly the fitness-evaluations. For very-fast fitness-evaluations, CMA-ES might be prohibitively too expensive.
- **Robustness requirement in highly unsmooth problems.** Evolutionary techniques are the only ones that outperform quasi-random-optimization in a stable manner even in the case of very unsmooth penalty-functions (see **-problems in the Table 2). The GA is not always the best optimizer, but in most cases it is at least better than random; we do believe that the well-known robustness of evolutionary algorithms, for the five meanings of robustness pointed out in section 1.1, are fully relevant for ADP.
- **A natural tool for generating bang-bang-efficient controlers.** In some cases (typically bang-bang problems) the LD-discretization introducing a bias towards the frontiers are (unsurprisingly) the best ones, but for other problems LD leads to the worst results of all techniques tested. This is not a trivial result, as this points out LD as a natural way of generating nearly bang-bang solutions, which depending on the number of function-evaluations allowed, samples the middle of the action space, and then the corners, and then covers the whole action space (what is probably a good "anytime" behavior). A posteriori, LD appears as a natural candidate for such problems, but this was not so obvious a priori.

REFERENCES

Auger, A., Jebalia, M., and Teytaud, O. (2005). Xse: quasi-random mutations for evolution strategies. In *Proceedings of EA'2005*, pages 12–21.

Bäck, T., Hoffmeister, F., and Schwefel, H.-P. (1991). A survey of evolution strategies. In Belew, R. K. and Booker, L. B., editors, *Proceedings of the* 4^{th} *International Conference on Genetic Algorithms*, pages 2–9. Morgan Kaufmann.

Bäck, T., Rudolph, G., and Schwefel, H.-P. (1993). Evolutionary programming and evolution strategies: Similarities and differences. In Fogel, D. B. and Atmar, W., editors, *Proceedings of the* 2^{nd} *Annual Conference on Evolutionary Programming*, pages 11–22. Evolutionary Programming Society.

Bäck, T. and Schütz, M. (1995). Evolution strategies for mixed-integer optimization of optical multilayer systems. In McDonnell, J. R., Reynolds, R. G., and Fogel, D. B., editors, *Proceedings of the* 4^{th} *Annual Conference on Evolutionary Programming*. MIT Press.

Bertsekas, D. (1995). *Dynamic Programming and Optimal Control, vols I and II*. Athena Scientific.

Bertsekas, D. and Tsitsiklis, J. (1996). Neuro-dynamic programming, athena scientific.

Beyer, H.-G. (2001). *The Theory of Evolutions Strategies*. Springer, Heidelberg.

Beyer, H.-G., Olhofer, M., and Sendhoff, B. (2004). On the impact of systematic noise on the evolutionary optimization performance - a sphere model analysis, genetic programming and evolvable machines, vol. 5, no. 4, pp. 327 360.

Broyden., C. G. (1970). The convergence of a class of double-rank minimization algorithms 2, the new algorithm. j. of the inst. for math. and applications, 6:222-231.

Byrd, R., Lu, P., Nocedal, J., and C.Zhu (1995). A limited memory algorithm for bound constrained optimization. *SIAM J. Scientific Computing, vol.16, no.5*.

Cervellera, C. and Muselli, M. (2003). A deterministic learning approach based on discrepancy. In *Proceedings of WIRN'03, pp53-60*.

Collobert, R. and Bengio, S. (2001). Svmtorch: Support vector machines for large-scale regression problems. *Journal of Machine Learning Research*, 1:143–160.

Conn, A., Scheinberg, K., and Toint, L. (1997). Recent progress in unconstrained nonlinear optimization without derivatives.

DeJong, K. A. (1992). Are genetic algorithms function optimizers ? In Manner, R. and Manderick, B., editors, *Proceedings of the 2nd Conference on Parallel Problems Solving from Nature*, pages 3–13. North Holland.

Fitzpatrick, J. and Grefenstette, J. (1988). Genetic algorithms in noisy environments, in machine learning: Special issue on genetic algorithms, p. langley, ed. dordrecht: Kluwer academic publishers, vol. 3, pp. 101 120.

Fletcher, R. (1970). A new approach to variable-metric algorithms. computer journal, 13:317-322.

Gagné, C. (2005). Openbeagle 3.1.0-alpha.

Gelly, S., Ruette, S., and Teytaud, O. (2006). Comparison-based algorithms: worst-case optimality, optimality w.r.t a bayesian prior, the intraclass-variance minimization in eda, and implementations with billiards. In *PPSN-BTP workshop*.

Goldfarb, D. (1970). A family of variable-metric algorithms derived by variational means. mathematics of computation, 24:23-26.

Hansen, N. and Ostermeier, A. (1996). Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaption. In *Proc. of the IEEE Conference on Evolutionary Computation (CEC 1996)*, pages 312–317. IEEE Press.

Hickernell, F. J. (1998). A generalized discrepancy and quadrature error bound. *Mathematics of Computation*, 67(221):299–322.

Hooke, R. and Jeeves, T. A. (1961). Direct search solution of numerical and statistical problems. *Journal of the ACM, Vol. 8, pp. 212-229*.

Jin, Y. and Branke, J. (2005). Evolutionary optimization in uncertain environments. a survey, ieee transactions on evolutionary computation, vol. 9, no. 3, pp. 303 317.

Kaupe, A. F. (1963). Algorithm 178: direct search. *Commun. ACM*, 6(6):313–314.

Keijzer, M., Merelo, J. J., Romero, G., and Schoenauer, M. (2001). Evolving objects: A general purpose evolutionary computation library. In *Artificial Evolution*, pages 231–244.

LaValle, S. M., Branicky, M. S., and Lindemann, S. R. (2004). On the relationship between classical grid search and probabilistic roadmaps. *I. J. Robotic Res.*, 23(7-8):673–692.

L'Ecuyer, P. and Lemieux, C. (2002). Recent advances in randomized quasi-monte carlo methods. pages 419–474.

Lindemann, S. R. and LaValle, S. M. (2003). Incremental low-discrepancy lattice methods for motion planning. In *Proceedings IEEE International Conference on Robotics and Automation*, pages 2920–2927.

Niederreiter, H. (1992). *Random Number Generation and Quasi-Monte Carlo Methods*. SIAM.

Owen, A. (2003). *Quasi-Monte Carlo Sampling, A Chapter on QMC for a SIGGRAPH 2003 course*.

Sendhoff, B., Beyer, H.-G., and Olhofer, M. (2004). The influence of stochastic quality functions on evolutionary search, in recent advances in simulated evolution and learning, ser. advances in natural computation, k. tan, m. lim, x. yao, and l. wang, eds. world scientific, pp 152-172.

Shanno, D. F. (1970.). Conditioning of quasi-newton methods for function minimization. mathematics of computation, 24:647-656.

Sloan, I. and Woźniakowski, H. (1998). When are quasi-Monte Carlo algorithms efficient for high dimensional integrals? *Journal of Complexity*, 14(1):1–33.

Sutton, R. and Barto, A. (1998). *Reinforcement learning: An introduction.* MIT Press., Cambridge, MA.

Tsutsui, S. (1999). A comparative study on the effects of adding perturbations to phenotypic parameters in genetic algorithms with a robust solution searching scheme, in proceedings of the 1999 ieee system, man, and cybernetics conference smc 99, vol. 3. ieee, pp. 585 591.

Tuffin, B. (1996). On the use of low discrepancy sequences in monte carlo methods. In *Technical Report 1060, I.R.I.S.A.*

Wasilkowski, G. and Wozniakowski, H. (1997). The exponent of discrepancy is at most 1.4778. *Math. Comp*, 66:1125–1132.

Wright, M. (1995). Direct search methods: Once scorned, now respectable. *Numerical Analysis (D. F. Griffiths and G. A. Watson, eds.), Pitman Research Notes in Mathematics*, pages 191–208. http://citeseer.ist.psu.edu/wright95direct.html.

Zhu, C., Byrd, R., P.Lu, and Nocedal, J. (1994). L-BFGS-B: a limited memory FORTRAN code for solving bound constrained optimization problems. *Technical Report, EECS Department, Northwestern University*.

Table 1: Summary of the characteristics of the benchmarks. The stock management problems theoretically lead to convex Bellman-functions, but their learnt counterparts are not convex. The ”arm” and ”away” problem deal with robot-hand-control; these two problems can be handled approximately (but not exactly) by bang-bang solutions. Walls and Multi-Agent problems are motion-control problems with hard penalties when hitting boundaries; the loss functions are very unsmooth.

Name	Nb of time steps	State space dimension (basic case)	Nb scenarios	Action space dimension (basic case)
Stock Management	30	4	9	4
Stock Management V2	30	4	9	4
Fast obstacle avoidance	20	2	0	1
Arm	30	3	50	3
Walls	20	2	0	1
Multi-agent	20	8	0	4
Away	40	2	2	2

Table 2: Experimental results. All stds are available at `http://www.lri.fr/~teytaud/sefordplong.pdf`. For the ”best algorithm” column, **bold** indicates 5% significance for the comparison with all other algorithms and *italic* indicates 5% significance for the comparison with all but one other algorithms. **y** holds for 10%-significance. Detailed results in `http://www.lri.fr/~teytaud/sefordplong.pdf` show that many comparisons are significant for larger families of algorithms, e.g. if we group GA and GANM, or if we compare algorithms pairwise. Problems with a star are problems for which bang-bang solutions are intuitively appealing; LD, which over-samples the frontiers, is a natural candidate for such problems. Problems with two stars are problems for which strongly discontinuous penalties can occur; the first meaning of robustness discussed in section 1.1 is fully relevant for these problems. Conclusions: 1. GA outperforms random and often QR. 2. For *-problems with nearly bang-bang solutions, LD is significantly better than random and QR in all but one case, and it is the best in 7 on 8 problems. It’s also in some cases the worst of all the tested techniques, and it outperforms random less often than QR or GA. LD therefore appears as a natural efficient tool for generating nearly bang-bang solutions. 3. In **-problems, GA and GANM are often the two best tools, with strong statistical significance; their robustness for various meanings cited in section 1.1 make them robust solutions for solving non-convex and very unsmooth problems with ADP. 4. Stock management problems (the two first problems) are very efficiently solved by CMA-ES, which is a good compromise between robustness and high-dimensional-efficiency, as soon as dimensionality increases.

Problem	Dim.	Best algo.	QR beats random	GA beats random ; QR	LBFGSBrestart beats random;QR	LD beats random;QR
Stock	4	**LDfff**	y	y;n	y ; n	y ; n
and	8	**EoCMA**	**y**	n;n	n ; n	n ; n
Demand	12	*EoCMA*	**y**	n;n	n ; n	n ; n
	16	**EoCMA**	n	n;n	n ; n	n ; n
Stock and	4	**LD**	**y**	**y**;y	**y**; **y**	**y**; **y**
Demand2	8	EoCMA	n	y;**y**	n ; **y**	y ; **y**
Avoidance	1	**HJ**	y	y;n	n ; n	**y**; y
Walls**	1	GA	y	y;y	y ; y	y ; y
Multi-agent**	4	**GA**	n	**y**;**y**	n ;n	n ; n
	8	**GANM**	y	**y**;**y**	n ;n	n ; n
	12	**LDfff**	**y**	**y**;**y**	n ;n	n ; n
	16	*GANM*	**y**	**y**;**y**	n ;n	y ; n
Arm*	3	LD	y	**y**;y	y ; y	**y**; **y**
	6	HJ	**y**	**y**;y	y ; y	**y**; **y**
	9	LD	**y**	**y**;n	y ; y	**y**; **y**
	12	**LD**	**y**	**y**;**y**	y ; y	**y**; **y**
Away*	2	LD	y	y;y	y ; n	**y**; y
	4	LD	y	**y**;y	y ; y	**y**; **y**
	6	LD	y	y;y	y ; y	**y**; **y**
	8	**LD**	y	y;y	y ; y	**y**; **y**
Total			17/20	17/20 ; 14/20	11/20 ; 10/20	14/20 ; 12/20

DETECTION OF THE NEED FOR A MODEL UPDATE IN STEEL MANUFACTURING

Heli Koskimäki (née Junno), Ilmari Juutilainen, Perttu Laurinen and Juha Röning
Intelligent Systems Group, University of Oulu, PO BOX 4500,90014 University of Oulu, Finland
{*heli.koskimaki,ilmari.juutilainen, perttu.laurinen, juha.roning*}*@ee.oulu.fi*

Keywords: Adaptive model update, similar past cases, error in neighborhood, process data.

Abstract: When new data are obtained or simply when time goes by, the prediction accuracy of models in use may decrease. However, the question is when prediction accuracy has dropped to a level where the model can be considered out of date and in need of updating. This article describes a method that was developed for detecting the need for a model update. The method is applied in the steel industry, and the model whose need of updating is under study is a regression model developed to model the yield strength of steel plates. It is used to plan process settings in steel plate product manufacturing. To decide on the need for updating, information from similar past cases was utilized by introducing a limit called an exception limit. The limit was used to indicate when a new observation was from an area of the model input space where the prediction errors of the model have been too high. Moreover, an additional limit was formed to indicate when too many exceedings of the exception limit have occurred within a certain time scale. These two limits were then used to decide when to update the model.

1 INTRODUCTION

At the Ruukki's steel works in Raahe, Finland, liquid steel is cast into steel slabs that are then rolled into steel plates. Many different variables and mechanisms affect the mechanical properties of the final steel plates. The desired specifications of the mechanical properties of the plates vary, and to fulfill the specifications, different treatments are required. Some of these treatments are complicated and expensive, so it is possible to optimize the process by predicting the mechanical properties beforehand on the basis of planned production settings (Khattree and Rao, 2003).

Regression models have been developed for Ruukki to help development engineers control mechanical properties such as yield strength, tensile strength, and elongation of the metal plates (Juutilainen and Röning, 2006). However, acquirement of new data and the passing of time decrease the reliability of the models, which can bring economical losses to the plant. For example, when mechanical properties required by the customer are not satisfied in qualification tests, the testing lot in question need to be reproduced. If also retesting gives unsatisfactory result the whole order has to be produced again. Because of the volumes produced in a steel mill, this can cause huge losses. Thus, updating of the models emerges as an important step in improving modelling in the long run. This study concerns the need to update the regression model developed to model the yield strength of steel plates.

In practice, because the model is used in advance to plan process settings and since the employees know well how to produce common steel plate products, modelling of rare and new events becomes the most important aspect. However, to make new or rarely manufactured products, a reliable model is needed. Thus, when comparing the improvement in the model's performance, rare events are emphasized.

In this study, model adaptation is approached by searching for the exact time when the performance of the model has decreased too much. In practice, model adaptation means retraining the model at optimally selected intervals. However, because the system has to adapt quickly to a new situation in order to

avoid losses to the plant, periodic retraining, used in many methods ((Haykin, 1999), (Yang et al., 2004)), is not considered the best approach. Moreover, there are also disadvantages if retraining is done unnecessarily. For example, extra work is needed to take a new model into use in the actual application environment. In the worst case, this can result in coding errors that affect the actual accuracy of the model.

Some other studies, for example (Gabrys et al., 2005), have considered model adaptation as the model's ability to learn behavior in areas from which information has not been acquired. In this study, adaptation of the model is considered to be the ability to react to time-dependent changes in the modelled causality. In spite of extensive literature searches, studies that would be comparable with the approach used in this article were not found. Thus, it can be assumed that the approach is new, at least in an actual industrial application.

2 DATA SET AND REGRESSION MODEL

The data for this study were collected from the Ruukki's steel works production database between July 2001 and April 2006. The whole data set consisted of approximately 250,000 observations. Information was gathered from element concentrations of actual ladle analyses, normalization indicators, rolling variables, steel plate thicknesses, and other process-related variables (Juutilainen et al., 2003). The observations were gathered during actual product manufacturing. The volumes of the products varied, but if there were more than 500 observations from one product, the product was considered a common product. Products with less than 50 observations were categorized as rare products.

The response variable used in the regression modelling was the Box-Cox-transformed yield strength of the steel plates. The Box-Cox transformation was selected to produce a Gaussian-distributed error term. The studied regression model was a link-linear model $y_i = \mu_i + \varepsilon_i$, where $\mu_i = f(x_i\beta)$ and ε_i are independently distributed Gaussian errors. The length of the parameter vector β was 130. In addition to 30 original input variables, the input vector x_i included 100 carefully chosen non-linear transformations of original input variables; for example, many of these transformations were products of two or three original inputs. The results are presented in the original (non-transformed) scale of the response variable.

3 NEIGHBORHOOD AND *APEN*

In this study the need for a model update was approached using information from previous cases, namely the average prediction errors of similar past cases. Thus, for every new observation, a neighborhood containing similar past cases was formed and an average prediction error inside the neighborhood was calculated.

The neighborhoods were defined using a Euclidian distance measure and the distance calculation was done only for previous observations to resemble the actual operating environment. The input variables were weighted using gradient-based scaling, so the weighting was relative to the importance of the variables in regression model (Juutilainen and Röning, ress). A numerical value of 3.5 was considered for the maximum distance inside which the neighboring observations were selected. The value was selected using prior knowledge of the input variable values. Thus, a significant difference in certain variable values with the defined weighting resulted in Euclidean distances of over 3.5. In addition to this, the maximum count of the selected neighbors was restricted to 500.

After the neighborhood for a new observation was defined, the average prediction error of the neighborhood (= *APEN*) was calculated as the distance-weighted mean of the prediction errors of observations belonging to the neighborhood:

$$APEN = \frac{\sum_{i=1}^{n}[(1-\frac{d_i}{max(d)})\cdot\widehat{\varepsilon_i}]}{\sum_{i=1}^{n}(1-\frac{d_i}{max(d)})}, \qquad (1)$$

where

- n = number of observations in a neighborhood,
- $\varepsilon(i)$ = the prediction error of ith observation of the neighborhood,
- d_i = the Euclidian distance from the new observation to ith observation of the neighborhood,
- $max(d)$ = the maximum allowed Euclidian distance between the new observation and the previous observations in the neighborhood (= 3.5).

4 STUDY

The method used to observe the need for a model update was to determine when the model's average prediction error in the neighborhood of a new observation (*APEN*) differed from zero too much compared with the amount of similar past cases. When there are plenty of accurately predicted similar past cases, the *APEN* is always near zero. When the amount of similar past cases decreases, the sensitivity of the *APEN* (in relation to measurement variation) increases, also in situations when the actual model would be accurate. In other words, the relationship between the sensitivity of the *APEN* and the number of neighbors is negatively correlated. The actual updating is also time-dependent, which means the model is updated when too many observations have an *APEN* value that differs significantly from zero within a certain time interval.

A limit, called the exception limit, was introduced to detect the need to update the model. The limit defines how high the absolute value of the average prediction error of a neighborhood ($= |APEN|$) has to be in relation to the size of the neighborhood before it can be considered an exception. This design was introduced to avoid possible sensitivity issues of the *APEN*. In practice, if the size of the neighborhood was 500 (the area is well known), prediction errors higher than 8 were defined as exceptions, while with a neighborhood whose size was 5, the error had to be over 50. The values of the prediction errors used were decided by relating them to the average predicted deviation, $\hat{\sigma}_i$ (≈ 14.4). The predicted deviations were acquired by using a regression model (Juutilainen and Röning, 2006). The limit is shown in Figure 1.

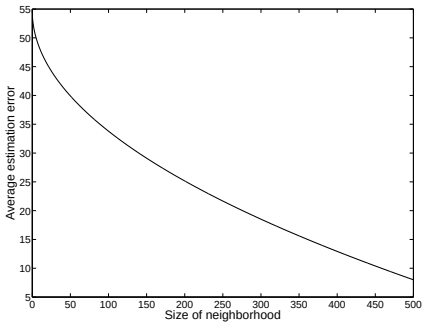


Figure 1: Exception limit.

A second limit, the update limit, was defined as being exceeded if 10 percent of the average prediction errors of the neighborhoods within a certain time interval exceeded the exception limit. The chosen interval was 1000 observations, which represents measurements from approximately one week of production. Thus, the model was retrained every time 100 of the preceding 1000 observations exceeded the exception limit.

The study was started by training the parameters of the regression model using the first 50,000 observations (approximately one year). After that the trained model was used to give the *APEN*s of new observations. The point where the update limit was exceeded the first time was located and the parameters of the model were updated using all the data acquired by then. The study was carried on by studying the reliability of the model after each update and repeating the steps iteratively until the whole data set was passed.

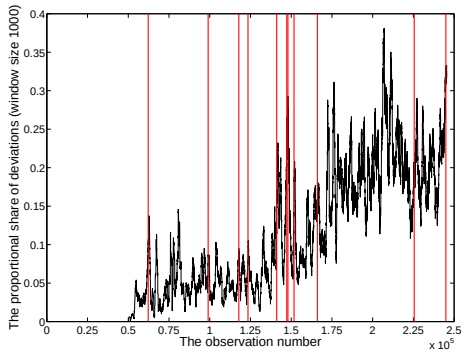
5 RESULTS

First the reliability of the model with and without updating was studied. Figure 2 shows the average prediction errors of the respective neighborhoods. Figure 2(a) shows the model's performance without updating. The straight line in the beginning represents the training data and the rest of the curve shows the proportional share of the exceedings of the exception limit, in other words, the percentage of *APEN*s that exceeded the exception limit. For example, it can be seen that at the end of the curve the *APEN* of every fourth new observation per week has exceeded the exception limit. The vertical lines mark the points of iteration where the update limit has been exceeded. On the other hand, Figure 2(b) presents the reliability of the model when the parameters are re-estimated at times indicated by the vertical lines. The curve represents the error rate of the re-trained model.

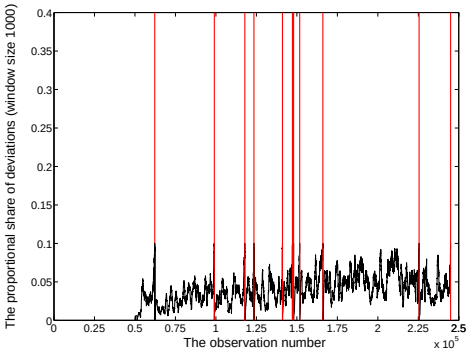
The positive effect of the updates on the average prediction errors of the neighborhoods can be clearly seen from Figure 2. Therefore, it is evident that the model's performance increases when the developed updating strategy is applied.

Although the reliability information of the model is very useful, the effect of the update on the actual prediction error was considered more important. Two different goodness criteria were used to compare the actual prediction errors of the updated models. Both criteria emphasize rare events, because that is the case when the model is needed the most. Both of them show the weighted average of the absolute prediction errors, with different weighting schemes.

In the first goodness criterion the idea is to emphasize all the products by approximately an equal amount. This means that in the calculation of the average prediction error, the weights of observations



(a) Without update.



(b) With update.

Figure 2: Reliability of the model.

belonging to products having more than 50 observations are decreased. The results of the goodness criterion are shown in Table 1. The step size indicates the length of the iteration step (comparable with the space between the vertical lines in Figure 2). The results are averages of the absolute prediction errors of the observations between each iteration step. Thus, the size of the data set used to calculate the average is the same as the step size. In addition to this, to compare the results, the prediction error averages are presented in three different cases: predicted with a newly updated model, with a model updated in the previous iteration step, and with a model that was not updated at all. The results show that, although the differences between the new model and the model from the previous iteration step are not big, the update improves the prediction in each of the steps. The benefit of the model update is obvious when the results of the updated model and the model without an update are compared.

The second goodness criterion was formed to take only rare events into account. Thus, only the cases

Table 1: Means of absolute prediction errors with the first goodness criterion.

Step size	With new model	With previous model	Without update
12228	-	-	-
36812	11.59	11.74	11.74
18826	10.47	11.00	10.69
5623	10.93	11.08	11.25
17636	12.48	12.57	12.97
6165	11.48	12.38	13.31
699	29.92	30.20	41.00
3765	11.79	12.21	13.96
14317	12.35	12.42	13.53
59432	12.22	12.39	12.87
19455	12.72	13.46	14.81
507	11.46	11.68	15.22
mean	**12.07**	**12.32**	**12.79**

where the neighborhood size is smaller than 50 are considered. The absolute prediction errors of these rare observations affect the average equal amount. The update proves its functionality in this scheme, also. The prediction errors are notably smaller when the model is updated (see Table 2).

Table 2: Means of absolute prediction errors with the second goodness criterion.

Step size	With new model	With previous model	Without update
12228	-	-	-
36812	14.41	14.63	14.63
18826	13.14	14.12	13.99
5623	12.94	13.11	13.10
17636	16.48	16.60	18.22
6165	12.60	13.90	14.71
699	84.86	85.86	115.44
3765	13.88	17.08	21.28
14317	16.23	16.21	22.09
59432	15.42	15.54	17.90
19455	26.42	27.88	43.87
507	15.99	15.18	16.04
mean	**15.67**	**16.04**	**18.10**

With this data set, determination of the need for a model update and the actual update process proved their efficiency. The number of iteration steps seems to be quite large, but like in Figure 2, the iteration steps get longer when more data is used to train the model. Thus, the amount of updates decreases as time goes on. However, the developed approach can also adapt to changes rapidly, when needed, as when new products are introduced or the production method of an existing product is changed. Finally, the benefits of this more intelligent updating procedure are obvious

in comparison with a dummy periodic update procedure (when the model is updated at one-year intervals, for example, the prediction error means of the whole data sets are 12.24 using criterion 1 and 16.34 using criterion 2, notably worse than the results achieved with our method, 12.07 and 15.67). The periodic procedure could not react to changes quickly or accurately enough, and in some cases it would react unnecessarily.

6 CONCLUSIONS

This paper described the development of a method for detecting the need for a model update in the steel industry. The prediction accuracy of regression models may decrease in the long run, and a model update at periodic intervals may not react to changes rapidly and accurately enough. Thus, there was a need for a reliable method for determining suitable times to update the model. Two limits were used to detect these update times, and the results appear promising. In addition, it is possible to rework the actual values of the limits to optimize the updating steps and improve the results before implementing the method in an actual application environment. Although the procedure was tested using a single data set, the extensiveness of the data set clearly proves the usability of the procedure. Nevertheless, the procedure will be validated when it is adapted also to regression models developed to model the tensile strength and elongation of metal plates.

In this study the model update was performed by using all the previously gathered data to define the regression model parameters. However, in the future the amount of data will increase, making it hard to use all the gathered data in an update. Thus, new methods for intelligent data selection are needed to form suitable training data. In addition, the model update could be developed into a direction where the input variables of the regression model can also be changed during the update.

REFERENCES

Gabrys, B., Leiviskä, K., and Strackeljan, J. (2005). *Do Smart Adaptive Systems Exist, A Best-Practice for Selection and Combination of Intelligent Methods*. Springer-Verlag, Berlin, Heidelberg.

Haykin, S. (1999). *Neural Networks, A Comprehensive Foundation*. Prentice Hall, Upper Saddle River, New Jersey.

Juutilainen, I. and Röning, J. (2006). Planning of strength margins using joint modelling of mean and dispersion. *Materials and Manufacturing Processes*, 21:367–373.

Juutilainen, I. and Röning, J. (2007, in press). A method for measuring distance from a training data set. *Communications in Statistics*.

Juutilainen, I., Röning, J., and Myllykoski, L. (2003). Modelling the strength of steel plates using regression analysis and neural networks. *Proceedings of International Conference on Computational Intelligence for Modelling, Control and Automation*, pages 681–691.

Khattree, R. and Rao, C., editors (2003). *Statistics in industry - Handbook of statistcs 22*. Elsevier.

Yang, M., Zhang, H., Fu, J., and Yan, F. (2004). A framework for adaptive anomaly detection based on support vector data description. *Lecture Notes in Computer Science, Network and Parallel Computing*, pages 443–450.

ROBUST ADAPTIVE WAVELET NEURAL NETWORK TO CONTROL A CLASS OF NONLINEAR SYSTEMS

A. Hussain, N. Essounbouli, A. Hamzaoui
CReSTIC, IUT de Troyes,9 rue de Québec, Troyes, France
{ayman.hussain,najib.essounbouli, abdelaziz.hamzaoui}@.univ-reims.fr

J. Zaytoon
CReSTIC, Uiversité de Reims Champagne-Ardenne, Reims, France
Janan.zaytoon@univ-reims.fr

Keywords: Adaptive control, adaptive wavelet neural network systems, adaptive identification, nonlinear systems, sliding mode control.

Abstract: This paper deals with the synthesis of a Wavelet Neural Network adaptive controller for a class of second order systems. Due to its fast convergence, the wavelet neural network is used to approximate the unknown system dynamics. The proposed approximator will be on-line adjusted according to the adaptation laws deduced from the stability analysis. To ensure the robustness of the closed loop system, a modified sliding mode control signal is used. In this work, variable sliding surface is considered to reduce the starting energy without deteriorating the tracking performances. Furthermore, the knowledge of the upper bounds of both the external disturbances and the approximation errors is not needed. The global stability of the closed loop system is guaranteed in the sense of Lyapunov. Finally, a simulation example is presented to illustrate the efficiency of the developed approach.

1 INTRODUCTION

In last decade, active research has been carried out in neural network control (Omidvar, 97) (Noriega, 98) (Lin, 98). The characteristics of fault tolerance, parallelism and learning suggest that they may be good candidates for implementing real-time adaptive control for nonlinear dynamical systems. It has been proven that an artificial neural network can approximate a wide range of nonlinear functions to any desired degree of accuracy under certain conditions (Omidvar, 97). It is generally understood that the selection of the neural network training algorithm plays an important role for most neural network applications. In the conventional gradient-descent-type weight adaptation, the sensitivity of the controlled system is required in the online training process (Lin, 98). However, it is difficult to acquire sensitivity information for unknown or highly nonlinear dynamics. Moreover, the local minimum of the performance index remains variable (Omidvar, 97). In practical control applications, it is desirable to have a systematic method for ensuring the stability, robustness, and performance properties of the overall system. Recently, several neural network control approaches have been proposed based on Lyapunov stability theory (Fabri, 96) (Farrell, 98) (Seshagiri, 00). One main advantage of these control schemes is that the adaptive laws were derived based on the Lyapunov synthesis method and therefore, the stability of the control system is guaranteed. However, some constraint conditions should be assumed in the control process, e.g., the approximation error, optimal parameter vectors, or higher order terms in a Taylor series expansion of the nonlinear control law are bounded. Also, the prior knowledge of the controlled system may be required, e.g., the external disturbance is bounded or all states of the controlled system are measurable. These requirements are not easy to satisfy in practical control applications. Recently, Wavelet Neural Networks (WNN) have became a very active subject in many scientific and engineering research areas (Zhang, 95) (Kostka, 00) (Lin, 03) (Ho, 05). The WNN have been proposed as a universal tool for functional approximation, which combine the capability of artificial neural networks in learning

and the capability of wavelet decomposition. The WNN allows resolving the conventional problem of poor convergence or even divergence encountered in other kinds of neural networks. It can also increase convergence speed (Delyon, 95) (Hsu, 06).

Sliding mode control is unique in its ability to achieve accurate, robust, decoupled tracking for a class of nonlinear time-varying systems in the presence of disturbances and parameter variations (Utkin, 77) (Salamci, 01). The tracking of the desired trajectory is achieved through two phases: an approach phase, where the system is controlled to attain a predefined sliding surface, and a sliding phase along the sliding surface. However, in order to deal with the presence of modelling imprecision and disturbances, the control law has to be discontinuous across the sliding surface. Since the implementation of the associated control switching is necessarily imperfect, this leads to chattering which involves high control activity and may excite high-frequency dynamics and can, therefore, damage the plant (Slotine, 91). To resolve this problem, many solutions have been proposed in the literature (Slotine, 91)- (Lin, 02) (Berstecher, 01) (Hwang, 01) (Lin, 95) (Wai, 04). In (Slotine, 91), a boundary-layer in the neighbouring of the sliding surface has been defined to obtain a continuous behaviour of the control signal across this surface. Based on the same idea, a fuzzy system has been used to define this boundary layer and to exploit the human knowledge (Lin, 02). To remove the discontinuity in the control signal, some approaches combining sliding mode control and classical controller using a fuzzy supervisor can be cited (Berstecher, 01) (Hwang, 01) (Lin, 95). These methods resolve the problem related to the chattering phenomenon. However, to design the switching signal assuring the approaching phase, the upper bounds of both the external disturbances and the structural uncertainties must be well known. To overcome these problems, the authors of (Hamzaoui, 04) (Wai, 04) have proposed an approximation of the switching signal by an adaptive fuzzy system to eliminate the chattering phenomenon without requiring any particular knowledge about the upper bounds of both approximation errors and external disturbances. Nevertheless, the global stability of the closed loop system in these approaches is guaranteed only for a good approximation level or for a particular choice of the initial values of the adjustable parameters.

This paper proposes a wavelet Neural Network Adaptive Control (WNNAC) for a class of second-order nonlinear, uncertain and perturbed systems; this controller combines the advantages of WNN identification and the robustness of sliding mode control. The control law is composed of two parts. The first one represents the WNN identifier that perform the online system dynamic function estimation. This identifier is adjusted according some adaptations laws deduced from the stability analysis. The second part of the control law represents the robust term which ensure the robustness of the closed loop system in the sense of sliding mode technique. This term is synthesised such that knowing of the upper bounds of the external disturbances is not required. Furthermore, the proposed control law uses a variable sliding surface to reduce the starting energy obtained by a classical sliding surface. The stability of the closed loop system is stated using the Lyapunov theory. To illustrate the efficiency of the proposed approach, a numerical simulation example is considered.

The paper is organised as follows: Section 2 illustrates the description of WNN networks. Section 3 is dedicated to the formulation and the investigation of the control problem. In section 4, we present the synthesis of the proposed controller whose design procedure is explained in section 5. To show the efficiency of the proposed approach, a simulation example is presented in section 6.

2 DESCRIPTION OF WAVELET NEURAL NETWORKS (WNNS)

Wavelet neural networks are special case of feed-forward neural networks. The main difference between the artificial neural networks (ANN) and WNN is that, in ANN the nonlinearities are approximated by superposition of sigmoid functions. However, in WNN, nonlinearities are approximated by superposition of wavelet functions (Oussar, 98). Similar to ANN, WNN are also shown to have universal approximation property (Yoo, 05) (Sureshbabu, 99).

Similar to the ANN, the WNN consists of an input layer, a hidden layer, and an output layer. The WNN model structure shown in Figure 1 consists of $'n'$ input neurons $(x_1, x_2, \ldots, x_n)$ in the input layer, equal to the number of input variables. The input neurons are connected to the next layer of neurons, called the hidden layer neurons which make use of wavelets as transformation functions. These neurons are termed as "wavelons". In this work, the Mexican hat (1) is used as a 'mother' wavelet ψ .

$$\psi(x) = (1 - x^2) \times \exp(-0.5x^2) \qquad (1)$$

Several daughters of wavelets ψ_j can be constructed by translating and dilating the mother wavelet ψ according to (2).

$$\psi_j(z) = \psi\left(\frac{\sigma_j - t_j}{d_j}\right) \tag{2}$$

where $\sigma_j = \sum_{i=1}^{n} v_{ij} x_i$, v_{ij} is input scaling vector while t_j, and d_j represent the translation and dilation factors of the wavelet.

For *i* and *j* represent the indices of input, hidden, and output layers respectively, the output from the hidden wavelon, H_j is given by (3).

$$H_j(x) = \sum_{i=1}^{n} \psi\left(\frac{\sigma_j - t_j}{d_j}\right) \tag{3}$$

This output is connected directly to the output layer neurons. The output layer usually consists of a linear output neuron. Mathematically, the final production obtained from figure 1 can be represented by (4).

$$Y_{output} = \underline{C}^T \Psi + \underline{A}^T \underline{X} + B \tag{4}$$

with $\underline{C}^T = [c_1 \ \cdots \ c_J]^T$, $\Psi = [H_1 \ \cdots H_J]$ and $\underline{A}^T = [a_1 \ \cdots a_n]$

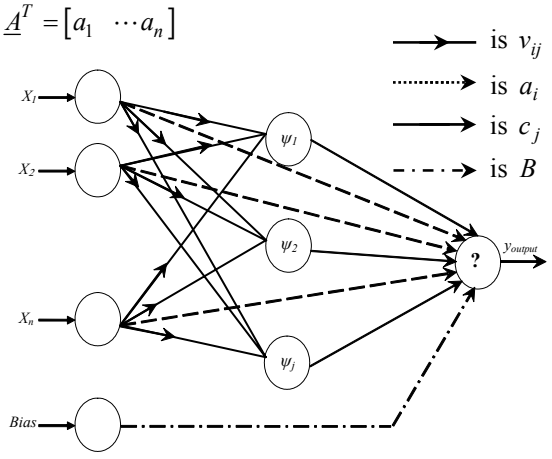


Figure 1: The structure of the used WNN.

The approximator in figure 1 includes three independent adaptable parts; constant, linear and non-linear given by B, $\underline{A}$ and $\underline{C}$ respectively. Thus, the approximator (4) is able to estimate efficiently all possible systems static or dynamics by managing learning rate of each part.

3 PROBLEM STATEMENT

Consider the following 2nd order system:

$$\ddot{y} = f(y, \dot{y}) + g(y, \dot{y})u + d \tag{5}$$

where f and g are two unknown continuous functions. u and y designate the input and output of the system respectively while d is an external disturbance assumed to be unknown but bounded. The input-output system (5) includes a large class of non-linear second-order systems likes Duffing oscillator and mass-spring-damper system (Bartoloni, 97) (Roup, 01) (Chang, 05).

In this work, we assume that the function $g(y, \dot{y})$ can be written as a sum of a known nominal term and an uncertain as follows

$$g(y, \dot{y}) = g_0(y, \dot{y}) + \delta_g(y, \dot{y})$$

In this case, equation (5) can be rewritten as:

$$\ddot{y} = f(y, \dot{y}) + g_0(y, \dot{y})u + \delta_d \tag{6}$$

where $\delta_d = \delta_g u + d$.

The objective of this work is to synthesize a robust controller, based on sliding mode, able to force the output of the system *y* to follow a bounded reference trajectory y_d under the constraint that all involved signals are bounded.

We denote the tracking error by $e = y_d - y$ and the sliding surface by:

$$s = \dot{e} + \lambda e \tag{7}$$

Since using a large value of the slop λ gives a fast system response. However, a too large value can lead to overshoot and even instability. From the other side, small value of λ results a slow system response. To overcome this problem, the slope can be adapted according to tracking error value $e(t)$ (Liu, 05). In this case, equation (7) becomes

$$s = \dot{e} + \lambda(e)e \tag{8}$$

where $\lambda(e) = \frac{k_\lambda}{|e| + \varepsilon_\lambda}$, k_λ is a given positive constant and ε_λ is a small positive scalar given by the designer.

Differentially according time of equation (8) gives

$$\begin{aligned} \dot{s}(t) &= \ddot{e}(t) + \lambda\dot{e}(t) + \dot{\lambda}e(t) \\ &= \ddot{y}_d - \ddot{y} + \lambda\dot{e}(t) + \dot{\lambda}e(t) \end{aligned} \tag{9}$$

Using (6), equation (9) becomes

$$\dot{s}(t) = \ddot{y}_d - f - g_0 u - \delta_d + \lambda \dot{e}(t) + \dot{\lambda} e(t) \tag{10}$$

$$\dot{s}(t) = -f - \delta_d + \dot{\lambda} e(t) - g_0 u + \ddot{y}_d + \lambda \dot{e}(t) \tag{11}$$

$$\dot{s}(t) = -F(y, \dot{y}, y_d) - g_0 u + \ddot{y}_d + \lambda \dot{e}(t) \tag{12}$$

where $F(y, \dot{y}, y_d) = f + \delta_d - \dot{\lambda} e(t)$.

In the next section, the problem of $F(y, \dot{y}, y_d)$ term estimation will be treated

4 THE WNN ADAPTIVE CONTROLLER SYNTHEISIS

It is worthy to say that if the system output and its time derivative converge to their reference signal, the unknown function $F(y, \dot{y}, y_d)$ goes to $F(y_d, \dot{y}_d)$. To approximate $F(y, \dot{y}, y_d)$ we can use a WNN in the form

$$\hat{F}(y_d, \dot{y}_d) = \underline{C}^T \psi + \underline{A}^T Y_d + B$$

with y_d and $\dot{y}_d$ as inputs (Chang, 05).

Consider the pre-assigned constraint regions of $\underline{C}$, $\underline{A}$ and B defined respectively as:

$$\Omega_C = \left\{ \underline{C} / \left\| \underline{C} \right\| \le M_c, M_c > 0 \right\}$$

$$\Omega_A = \left\{ \underline{A} / \left\| \underline{A} \right\| \le M_A, M_A > 0 \right\}$$

$$\Omega_B = \left\{ B / \left\| B \right\| \le M_B, M_B > 0 \right\}$$

According to the approximation theorem, there exists a finite optimal value of $F(y_d, \dot{y}_d)$ noted by:

$$\hat{F}^*(y_d, \dot{y}_d) = \underline{C}^{*T} \psi + \underline{A}^{*T} Y_d + B^*$$

($\underline{C} \in \Omega_C$, $\underline{A} \in \Omega_A$ and $B \in \Omega_B$) such that:

$$\delta_F = F(y_d, \dot{y}_d) - \hat{F}^*(y_d, \dot{y}_d)$$

hence, equation (12) can be rewritten as:

$$\begin{aligned} \dot{s}(t) = &-\hat{F}^*(y_d, \dot{y}_d) + \delta_F - g_0 u + \ddot{y}_d \\ &+ \lambda \dot{e}(t) + \delta_d \end{aligned} \tag{13}$$

To guarantee the global stability of the closed loop system and the convergence of the tracking error to zero, we propose the following control law:

$$u(t) = g_0^{-1} \left[-\hat{F}(y_d, \dot{y}_d) + \ddot{y}_d + \lambda \dot{e} + \frac{s}{\rho^2} \right] \tag{14}$$

Substituting (14) in (13) yields to:

$$\begin{aligned} \dot{s}(t) = &-\hat{F}^*(y_d, \dot{y}_d) + \hat{F}(y_d, \dot{y}_d) - \ddot{y}_d \\ &- \lambda \dot{e} + \ddot{y}_d + \lambda \dot{e} + \delta_F + \delta_d - \frac{s}{\rho^2} \end{aligned} \tag{15}$$

which can be rewritten as

$$\begin{aligned} \dot{s}(t) = &\tilde{\underline{C}}^T \psi + \tilde{\underline{A}}^T y_d + \tilde{B} \\ &+ \delta_F + \delta_d - \frac{s}{\rho^2} \end{aligned} \tag{16}$$

To study the stability of the closed loop system and to find the adaptation laws for the adjustable parameters, we consider the following Lyapunov function:

$$v(t) = \frac{1}{2} s^2 + \frac{1}{2\gamma_C} \tilde{\underline{C}}^T \tilde{\underline{C}} \frac{1}{2\gamma_A} \tilde{\underline{A}}^T \tilde{\underline{A}} + \frac{1}{2\gamma_B} \tilde{B}^2 \tag{17}$$

The time derivative of $v(t)$ is given by:

$$\dot{v}(t) = s\dot{s} + \frac{1}{\gamma_C} \tilde{\underline{C}}^T \dot{\tilde{\underline{C}}} \frac{1}{\gamma_A} \tilde{\underline{A}}^T \dot{\tilde{\underline{A}}} + \frac{1}{\gamma_B} \tilde{B} \dot{\tilde{B}} \tag{18}$$

Substituting (16) in (18) gives:

$$\begin{aligned} \dot{v}(t) = &\, s \left(\tilde{\underline{C}}^T \psi + \tilde{\underline{A}}^T y_d + \tilde{B} + \delta_F + \delta_d - \frac{s}{\rho^2} \right) \\ &+ \frac{1}{\gamma_C} \tilde{\underline{C}}^T \dot{\tilde{\underline{C}}} \frac{1}{\gamma_A} \tilde{\underline{A}}^T \dot{\tilde{\underline{A}}} + \frac{1}{\gamma_B} \tilde{B} \dot{\tilde{B}} \\ \dot{v}(t) = &\, s\delta_F + s\delta_d - \frac{s^2}{\rho^2} + \frac{1}{\gamma_C} \tilde{\underline{C}}^T \left(\dot{C} + \gamma_C s \psi \right) \\ &+ \frac{1}{\gamma_A} \tilde{\underline{A}}^T \left(\dot{\underline{A}} + \gamma_A s y_d \right) + \frac{1}{\gamma_B} \tilde{B} \left(\dot{B} + \gamma_B s \right) \end{aligned} \tag{19}$$

Choosing the following adaptation law

$$\dot{\underline{C}} = -\gamma_C s \psi \tag{20}$$

$$\dot{\underline{A}} = -\gamma_A s y_d \tag{21}$$

$$\dot{B} = -\gamma_B s \tag{22}$$

leads to:

$$\dot{v}(t) = s\delta_F + s\delta_d - \frac{s^2}{\rho^2} \tag{23}$$

$$\begin{aligned} \dot{v}(t) = &-\frac{s^2}{4\rho^2} + 2\frac{s}{2\rho} \rho \delta_F - \frac{s^2}{4\rho^2} \\ &+ 2\frac{s}{2\rho} \delta_d - \frac{s^2}{2\rho^2} \end{aligned} \tag{24}$$

$$\dot{v}(t) = -\left(\frac{s}{\rho} - \rho\delta_F\right)^2 - \left(\frac{s}{\rho} - \rho\delta_d\right)^2 + \rho^2\delta_F^2 + \rho^2\delta_d^2 - \frac{s^2}{2\rho^2} \quad (25)$$

$$\dot{v}(t) \le -\frac{s^2}{2\rho^2} + \rho^2\delta_F + \rho^2\delta_d^2 \quad (26)$$

Let $\Omega_s = \left\{\frac{s}{\|s\|} \le \mu,\ \ \mu > 0\right\}$.

According to the regions Ω_S, Ω_C, Ω_A and Ω_B, there is a sufficient large constant $V_{\max}$ such that $V_{\max} \le \max_{s\in\Omega_S, C\in\Omega_C, A\in\Omega_A, B\in\Omega_B} v(t)$ (Chang-05).

Afterward, we can define

$$\Omega_{\dot{e}} = \left\{\frac{\dot{e}}{\|\dot{e}\|} \le \sqrt{2V_{\max}}\right\}$$

and

$$\Omega_e = \left\{\frac{e}{\|e\|} \le \sqrt{\frac{2\varepsilon_\lambda^2 V_{\max}}{k_\lambda^2}}\right\}$$

In the case where the sliding surface is outside the region Ω_S, for choosing

$$\frac{1}{\rho^4} > \frac{\delta_F^2 + \delta_d^2}{\mu^2}$$

there exists a positive constant ζ such that:

$$\dot{v}(t) \le -\zeta s^2 \quad (27)$$

From the definition of the constraint region, we have $v(0) \le V_{\max}$. From the inequality (23), we obtain $v(t) \le v(0) \le V_{\max}$ which implies that $s \in \Omega_s$ for all $t \ge 0$ and therefore $e \in \Omega_e$ and $\dot{e} \in \Omega_{\dot{e}}$ (Chang-05). Hence, $\Omega_s \times \Omega_C \times \Omega_A \times \Omega_B$ is an invariant set, we can conclude that all the variables are bounded. Since the Lyapunov function is negative outside the constraint set Ω_s, then sliding surface is Uniformly Ultimately Bounded (UBB) and hence the tracking error is also UBB.

By integrating the above inequality between *0* and *T*, we obtain:

$$v(T) - v(0) \le -\int_0^T \frac{s^2}{2\rho^2}dt + \int_0^T \rho^2\delta_F dt + \int_0^T \rho^2\delta_d^2 dt$$

$$\int_0^T \frac{s^2}{2\rho^2}dt \le v(0) - v(T) + \int_0^T \rho^2\delta_F dt + \int_0^T \rho^2\delta_d^2 dt \quad (28)$$

Since $v(T) \ge 0$, we have

$$\int_0^T \frac{s^2}{2\rho^2}dt \le \int_0^T \rho^2\delta_F dt + \int_0^T \rho^2\delta_d^2 dt$$

or

$$\int_0^T s^2 dt \le 4\rho^2\int_0^T \delta_F dt + 4\rho^2\int_0^T \delta_d^2 dt \quad (29)$$

Using the Barbalat's lemma (Wang, 94), one can see that the sliding surface converges asymptotically to zero despite the presence of external disturbances. Hence, the sliding surface is attractive, i.e., if the system attains the surface, it remains and converges toward the origin as demonstrated in (Utkin, 99).

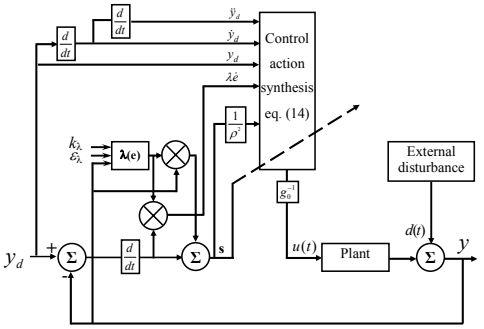


Figure 2: Control scheme of the proposed approach.

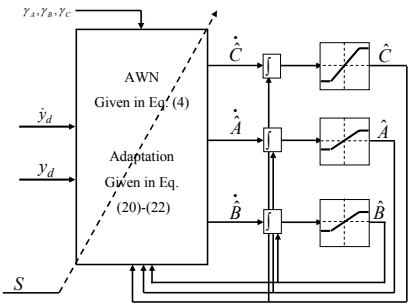


Figure 3: Adaptive Wavelet Network illustration.

5 DESIGN PROCEDURE

The control scheme of the proposed approach and the adaptive WNN block are given respectively by figures 2 and 3. The control law (14) can be synthesised according to the following steps:

Step 1: define the interval range for each input $(y_d, \dot{y}_d)$ and the corresponding wavelet parameters (dilation and translation) such that input range is covered uniformly. To adapt on-line the adjustable parameters $\underline{A}$, B and $\underline{C}$, we define the learning

rates γ_A, γ_B and γ_C. To increase approximator nonlinearity, we choose γ_C bigger than γ_A and γ_B.

Step 2: specify the parameters k_λ and ε_λ to calculate the sliding surface given in (8). To deduce the control law (14), we choose ρ. This value represents a trade-off between the initial starting energy and the time response in transient state.

Step 3: update the adjustable parameters of the WNN according to equation (20), (21) and (22).

6 SIMULATION EXAMPLE

In order to validate the proposed controller, the Wing-Rock Motion (WRM) system is considered in simulation. Indeed, some combat aircrafts often operate at subsonic speeds and high angles of attack. These aircrafts may become unstable due to oscillation, mainly a rolling motion known as WRM (Lan, 95) (Lin, 02). the dynamics of WRM system can be described in a state variable by:

$$\ddot{y} = b_0 + b_1 y + b_2 \dot{y} + b_3 |y| \dot{y} + b_4 |\dot{y}| \dot{y} + b_5 y^3 + u + d \quad (30)$$

Where y is the roll angle, u is the control action and d is the external disturbance. The parameters related to $b_i (i = 1,...,5)$ are the aerodynamic parameters given by:

$b_0 = 0$, $b_1 = -.01859521$, $b_2 = .015162375$, $b_3 = .06245153$, $b_4 = .00954708$ and $b_5 = .02145291$.

According to the design procedure given in section 5, we will define at first the AWNN. Indeed if we consider that $y_d = \cos(t)$ being the desired reference trajectory, then the interval range of the desired output (as well as the actual system output in case of perfect tracking) will belongs always to the closed interval [-1, 1]. For the purpose of reliability and to give some relaxation to our controller, the interval $[-1.5,1.5]$ is considered as a universe of discourse for both of input and hidden layers in the WNN approximator. The number of wavelons (wavelet neurons) components used in the hidden layer are four which is enough for covering the interval [-1.5, 1.5] adequately. For a WNN approximator, the number of waveleons components used in the hidden layer depends on the network input's interval and dynamic complexity for the system to be approximated. Translation parameters selection is considered in a way that guarantees the uniform covering of the [-1.5, 1.5] interval simply using linspace Matlab© instruction. Dilation parameters specify the intersection amount between wavelons activation function (daughter function) which is chosen to be 0.5 such that the horizontal axes $y = 0.5$ contains these intersection points. The adjustable parameters $\underline{A}$, B and $\underline{C}$ are initialized to zero such that random initialization case is avoided since it doesn't gives neither same initials nor same training speed. Alternatively, these parameters might be chosen through some trials to achieve favourable transient control performance. For constants γ_A, γ_B and γ_C corresponding to the learning rates, it is important to recognize there effects to approximate accurately the nonlinear system and to avoid masking nonlinear property in the structure shown in figure 1. For this, it is better choosing γ_C much bigger than γ_A and γ_B. In our example, the values $\gamma_c = 10$, $\gamma_B = 0.05$ and $\gamma_A = 0.05$ have been considered. According to the second step in the design procedure, the variable sliding mode may be achieved through variation in surface slope (λ). For this we choose $k_\lambda = 20$ and $\varepsilon_\lambda = 0.5$.

Several simulations have been done and figures 4 to 6 show the results obtained for $\rho = 0.5$ where the system is subjected to external disturbance with $d = 0.3\sin(2t)$. Figures 4 to 6 give the simulation results for two different initial positions. We remark that the system attains the reference trajectory rapidly despite that the initial condition is so far from the reference one. Short response time reflects the good convergence of WNN. Furthermore, figure 6 shows the elimination of the chattering phenomenon and the absence of the abrupt variations appearing in classical sliding mode control. Comparing the obtained results with those in the case of linear sliding surface (λ constant), the proposed approach guarantees the same tracking performances with 40% initial control action reduction.

7 CONCLUSION

In this paper a robust adaptive wavelet neural network to control a class of nonlinear systems was presented. The combination of WNN and sliding mode control allows to develop a robust controller to guarantee the good tracking performances and the closed loop system stability. Considering a variable sliding surface reduces the starting energy without deteriorating the tracking performances.

Furthermore, no knowledge about of the upper bounds of both the external disturbances and the approximation errors is required to synthesis the control law. Simulation results have been presented to show the efficiency of the proposed approach.
Current work is dedicated to the generalisation of this method to n^{th} multi-input multi-output systems.

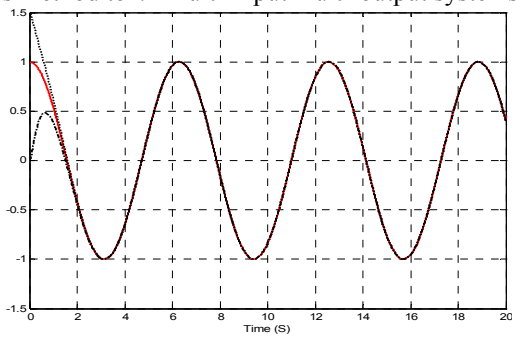


Figure 4: Evolution of the system output and its reference signal *(-:y_d, ..: y(t) with y(0)=1.5; -.-: y(t) with y(0)=0)*.

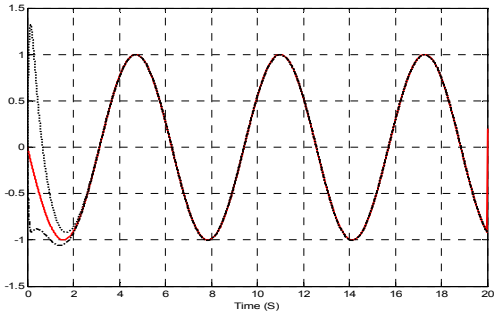


Figure 5: Evolution of the state $\dot{y}$ and its reference signal *(-: $\dot{y}_d$, ..: $\dot{y}$ with y(0)=1.5; -.-: $\dot{y}$ with y(0)=0)*.

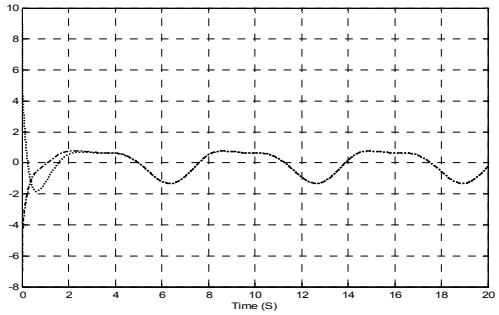


Figure 6: The control signal applied to the system *(...: u(t) with y(0)=1.5; -.-: u(t) with y(0)=0)*.

REFERENCES

Antsaklis, P. J., Jun. 1995. vol. 15, Issue 3, pp. 5 - 7. In *Intelligent Learning Control IEEE Control Syst. Mag.* IEEE Press.

Astrom, K. J., Wittenmark, B., 1995. In *Adaptive Control*. Addison-Wesley, 2nd edition.

Bartoloni, G., Ferrara, A., Usai, E., 1997. vol. 33 pp. 2203-2212. In *Automatica, Output tracking control of uncertain nonlinear second-order systems*. Elsevier.

Berstecher, R.G, Palm, R., Unbehauen, H.D., 2001. vol. 48, pp. 18-31. In *An Adaptive Fuzzy Sliding Mode Controller*, *IEEE Trans. on Indus. Electr.*, IEEE press.

Broderick, R. L., 2005. vol. 1, pp. 6.C.2-01-10. In *daptive verification for an on-line learning neural-based flight control system*, *the 24th Digital Avionics Systems Conference*, IEEE Press.

Chang, Y. C., Yen, H. M., 2005. vol. 35, pp. 1311-1316. In *Adaptive output Feedback control for a class of uncertain nonlinear systems using neural networks, IEEE Trans. Systems, Man, & Cybernetic*. IEEE Press.

Chang, Y.-C., 2005. part B: cybernetics vol. 35, no. 6 pp.1108- 1119. In *Intelligent Robust Control for Uncertain Nonlinear Time-Varying Systems and Its Application to Robotic Systems, IEEE Trans. on syst. Man & Cyber.,* IEEE Press.

Delyon, B., Juditsky, A., Benveniste, A., 1995. vol. 6, no. 2, pp.332–348. In *Accuracy analysis for wavelet approximations, IEEE Trans. Neural Networks.* IEEE Press.

Fabri, S., Kadirkamanathan, V., Sept. 1996. vol. 7, pp. 1151–1167, *Dynamic structure neural networks for stable adaptive control of nonlinear systems, IEEE Trans. Neural Networks* IEEE press.

Farrell, J., Baker, W., 1993. pp. 237-262. In *Learning control systems*, In Passino, K. M., Antsaklis, P. J. (Eds.), *An Introduction to Intelligent and Autonomous Control*. Kluwer Academic Publishers.

Farrell, J. A., Sept. 1998. vol. 9, pp.1008–1020 *Stability and approximator convergence in nonparametric nonlinear adaptive control," IEEE Trans. Neural Networks.* IEEE press.

Grman, J., Ravas, R., Syrová, L., 2001. vol. 1, no. 1, pp. 25–28. In *Application of Neural Networks in Multifrequency Eddy-Current Testing*, *Measurement Science Review*. Slovak Academy of Sciences.

Hamzaoui, A., Essounbouli, N., Zaytoon, J., 2004. vol. 218 (4), pp. 287-298. In *Fuzzy sliding mode control with fuzzy switching function for nonlinear uncertain MIMO Systems*, *Journal of Systems and Control Engineering*.

Ho, D. W. C., Li, J., Niu, Y., 2005. vol. 16, no. 3, pp. 625-635. *Adaptive Neural Control for a Class of Nonlinearly Parametric Time-Delay Systems, IEEE Trans. on Neural Networks,* IEEE Press.

Hsu, C.F., Lin, C.M., Lee, T.T., 2006. vol. 17, no. 5, pp. 1175-1183. In *Wavelet Adaptive Backstepping Control*

for a Class of Nonlinear Systems, IEEE Trans. On Neural Networks, IEEE Press.

Huang, S. N., Tan, K. K,. Lee, T. H., 2002. vol. 38, no. 2, pp. 227–233. In *Automatica, Adaptive motion control using neural network approximations*, Elsevier.

Hwang, C.-L., Kuo, C.-Y., 2001. vol. 9 (2), pp. 238-252. In *A stable adaptive fuzzy sliding-mode control for affine nonlinear systems with application to four-bar linkage systems*, *IEEE Trans. On Fuzzy Syst*. IEEE press.

Kostka, P., Tkacz, E. J., Nawrat, Z., Malota, Z., 2000. pp. 2463-2465. In *An application of wavelet neural networks for heart valve prostheses characteristic, Proceedings of the 22nd Annual EMBS International Conference*, IEEE Press.

Lan, C. E., Chen, Y., Lin, K. J., 1995. vol. 32, pp. 905–910. In *Experimental and analytical investigations of transonic limit-cycle oscillations of a flaperon, Journal Aircraft*, American Institute of Aeronautics and Astronautics.

Lin, C. M., Hsu, C. F., 2002. vol. 25, pp. 1163–1165,2002. In *Recurrent neural network adaptive control of wing rock motion*, *Journal of guidance, control, and dynamics*. American Institute of Aeronautics and Astronautics.

Lin, S.C., Chen, Y.Y., Mar. 1995. vol. 3, pp. 1103-1110. In *A GA-based fuzzy controller with sliding mode*, *Proc. IEEE Int.Conf. on Fuzzy Systems*. IEEE Press.

Lin, F. J., Hwang, W. J., Wai, R. J., 1998. vol. 145, no. 2, pp. 105–110, *Ultrasonic motor servo drive with on-line trained neural network model-following controller, Proc.IEE—Elect. Power Applicat*. . IEEE press.

Lin, W.-S., Chen, C.-S., 2002. vol. 149, pp. 193-202. In *Robust adaptive sliding mode control using fuzzy modelling for a class of uncertain MIMO nonlinear systems*, *IEE Proceedings: Control Theory and Applications*, IEEE Press.

Lin, F.J., Wai, R. J., Chen, M. P., 2003. vol. 20, no. 6, pp. 686-698. In *Wavelet Neural Network Control for Linear Ultrasonic Motor Drive via Adaptive Sliding-Mode Technique, IEEE Trans. On Ultrasonics, Ferroelectrics & Frequency Control*, IEEE Press.

Liu, Z. L., Nov. 2005. vol. 152, no. 6, pp. 615- 620 *Reinforcement adaptive fuzzy control of wing rock phenomena, IEE proc. Control theory appl.* IEE press.

Narendra, K. S., Parthasarathy, K., Mar. 1990. vol. 1, no. 1, pp. 4–27, 1990. In *Identification and control of dynamic systems using neural networks, IEEE Trans. Neural Net.* IEEE press.

Noriega, J. R., Wang, H, Jan. 1998. vol. 9, pp. 27–34.*A direct adaptive neural-network control for unknown nonlinear systems and its application, IEEE Trans. Neural Network,* IEEE Press.Omidvar, O., Elliott, D. L., 1997. *Neural Systems for Control*. New York: Academic.

Omidvar, O., Elliott, D. L., 1997. *Neural Systems for Control*. New York: Academic.

Oussar, Y., Rivals, I., Personnaz, L., Dreyfus, G., 1998. vol. 20, pp. 173-188, 1998. In *Training wavelet networks for nonlinear dynamic input-output modeling, Neurocomputing*, Elsevier

Psillakis, H. E., Alexandridis, A. T., Mar. 2006. vol. 153, No. 2. In *Adaptive neural motion control of n-link robot manipulators subject to unknown disturbances and stochastic perturbations. IEE Proc. Control Theory Appl.* IEE press.

Roup, A. V., Bernstein, D. S., 2001. vol. 46, pp. 1821-1825. In *Adaptive stabilization of a class of non-linear systems with parametric uncertainties*, *IEEE Trans. Automatic Control,* IEEE Press.

Salamci, M. U., Ozgoren, M. K., 2001. vol. 23, no. 4, pp. 719-727. *Sliding mode control with optimal sliding surfaces for missile autopilot design*, *Journal of Guidance, Control and Dynamics*, , 2001.

Sastry, S., Bodson, M., 1989. In *Adaptive Control: Stability, Convergence, and Robustness*. Prentice-Hall.

Seshagiri, S., Khalil, H. K., Jan. 2000. vol. 11, pp. 69–79, *Output feedback control of nonlinear systems using RBF neural network," IEEE Trans. Neural Networks* IEEE press.

Slotine, J.-J. E., Li, W., 1991. In *Applied Nonlinear Control*. Prentice-Hall.

Sureshbabu, N., Farrell, J. A., Feb. 1999. vol. 44, no. 2, pp.412-417. In *Wavelet-Based System Identification for Nonlinear Control, IEEE Transactions on automatic control,*IEEE Press.

Utkin, V.I., 1977. vol. 22, pp. 212-222.Variable structure systems with sliding modes," *IEEE Trans. Automatic Control*, IEEE Press.

Utkin, V. I., 1999. vol. 22, pp. 212-222. In *Variable structure systems with sliding modes, IEEE Trans. Automatic Control*, IEEE Press.

Wai, R.J., Lin, C. M., Hsu, C.F., 2004. vol. 143, pp. 295-310. In *Adaptive fuzzy sliding mode control for electrical servo drive*, *Fuzzy Sets & Systems*, Elsevier.

Wang, L.-X., 1994. In *Adaptive fuzzy systems and control*, Prentice-Hall.

Wong, L.K., Leung, F.H.F., Tam, P.K.S., 1998. vol. 1, pp. 296-301. In *Combination of sliding mode controller and PI controller Using fuzzy logic controller*, *IEEE Int. Conf. on Fuzzy Systems*. IEEE Press.

Yoo, S. J., Park, J. B., Choi, Y. H. Mar. 2005. vol. 3, no. 1, pp. 43–55. In *Stable Predictive Control of Chaotic Systems Using Self-Recurrent Wavelet Neural Network, International Journal of Control, Automation, and Systems*, IEEE Press.

Zhang, Q., Walter, G. G., Miao, Y., Lee, W. N. W., Jun. 1995. vol. 43, pp. 1485-1496. In *Wavelet neural networks for function learning, IEEE Trans. Signal Proc.,* IEEE Press.

A MULTI CRITERIA EVALUATION OVER A FINITE SCALE FOR MAINTENANCE ACTIVITIES OF A MOTORWAY OPERATOR

Céline Sanchez [(1)], Jacky Montmain[(2)], Marc Vinches[(2)] and Brigitte Mahieu[(1)]
[(1)] *Service Structure Viabilité Sécurité, Société des Autoroutes Estérel Côtes d'Azur Provence Alpes*
avenue de cannes, 06211 Mandelieu Cedex, France
cesanchez@escota.net, bmahieu@escota.net
[(2)] *Ecole des Mines d'Alès,6 avenue de Clavières, 30319 Alès Cedex, France*
jacky.montmain@ema.fr, marc.vinches@ema.fr

Keywords: Multi-criteria decision-making, Multi-criteria aggregation, Finite scale, Decision support system, Motorway infrastructure.

Abstract: The Escota Company aims at the formalization and improvement of the decisional process for preventive maintenance in a multi criteria (MC) environment. According to available pieces of knowledge on the infrastructure condition, operations are to be evaluated with regards to (w.r.t.) technical but also to conformity, security and financial criteria. This MC evaluation is modelled as the aggregation of partial scores attributed to an operation w.r.t. a given set of n criteria. The scores are expressed over a finite scale which can cause some troubles when no attention is paid to the aggregation procedure. This paper deals with the consistency of the evaluation process, where scores are expressed as labels by Escota's experts, whereas the aggregation model is supposed to deal with numerical values and cardinal scales. We try to analyse this curious but common apparent paradox in MC evaluation when engineering contexts are concerned. A robustness study of the evaluation process concludes this paper.

1 ESCOTA DECISION PROCESS

1.1 Context

The Escota Company, founded in 1956, is the leading operator of toll motorways in France. Due to its integration into the Provence-Alpes-Côte d'Azur region, Escota is committed, as every motorway operator, to a sustainable development approach, including the social, economic and environmental aspects of its activities. Every year, specific initiatives are undertaken, or repeated, to include the motorway network in a sustainable development approach. Within this scope, the Escota Company aims at the formalization and improvement of the decisional process for preventive maintenance and property management in a multi actors and multi criteria (MC) environment. These decisions concern upkeep, improvement and upgrading operations, involving technical, conformity, security or financial criteria. The operations are related to operating domains such as constructive works, carriageways, vertical roadsigns and carriageway markings, buildings, prevention of fire risks, open spaces… Managing such a complex infrastructure necessitates a dynamic Information Processing System (IPS) to facilitate the way decision-makers use their reasoning capabilities through adequate information processing procedure.

1.2 Valuation of the Infrastructure Condition

Periodic inspections are performed to detect and measure, as early as possible, any malfunction symptoms affecting an element of the infrastructure (EI). The expert in charge of an operating domain then analyses the technical diagnosis relative to the EI. He evaluates the situation seriousness in terms of technical risk analyses. This evaluation relies on a specific set of *n* criteria relative to his domain. An aggregation with a weighted arithmetic mean (WAM) is then performed to assess a global degree of emergency to the corresponding maintenance operation. This evaluation is then submitted to the official in charge of the operating network. This

latter coordinates the experts' needs and demands for operation planning purposes.

This paper deals more particularly with the MC evaluation process by the expert of an operating domain, i.e. the affectation of an emergency degree to an operation. There exist several methods to identify and perform aggregation process with a WAM. The Analytic Hierarchical Process, AHP, is probably the most famous one in industry (Saaty, 1980). However, because it explicitly guarantees the consistency between the commensurable scales it aggregates and the WAM operator it identifies, the Measuring Attractiveness by a Categorical Based Evaluation TecHnique method, MACBETH, has got recent successes (Bana e Costa, 1994)(Clivillé, 2004). In our application, MACBETH is first used to build the valuation scale associated to each emergency criterion of a domain. It is then applied to determine the WAM parameters.

Furthermore, the way experts give their assessment in natural language raises another problem (Jullien, 2006). These labels are commonly converted into numerical values to perform the aggregation process. No particular attention is generally paid to this "translation". However the consequences over the aggregation results are damageable. In civil engineering, the culture of numbers is strongly developed. People commonly manipulate symbolic labels but may convert them into more or less arbitrary numerical values when necessary without further care. This cultural viewpoint explains why an aggregation operator is generally preferred to a rule base whereas appraisals are expressed in terms of symbolic labels (Jullien, 2006). A completely symbolic evaluation over finite scales could be envisaged (Grabisch, 2006).

Let us illustrate the scales problem with the following example. Let us suppose that the semantic universe of an expert w.r.t. the seriousness of a symptom is: {*insignificant, serious, alarming*}. We can imagine that a corresponding possible set of discrete numerical values (in [0; 1]) could be: {0; 0.5; 1}. There are several assumptions behind this translation concerning the nature of the scale. This point will be discussed later. Let us just note here that the numerical values are commonly chosen equidistant. Now let us consider another semantic universe: {*insignificant, minor, alarming*}. This time, the associated set of numerical values {0; 0.5; 1} intuitively appears more questionable. The expert should prefer {0; 0.25; 1}. When seriousness degrees of several symptoms are to be aggregated, the result of the WAM aggregation strongly depends on the choice of the set of numerical values. Furthermore, in any case, the numerical WAM value does not necessary belong to {0; 0.5; 1} or {0; 0.25; 1}. It must then be converted into the convenient label in return.

The way labels are converted into numerical values (and back) coupled to the commensurability of the scales of the dimensions to be aggregated can entail serious problems when aggregating without any care. In this paper, we propose a methodology to build finite partial valuation scales consistently with WAM aggregation.

The paper is organized as follows. Some considerations are given about the way continuous cardinal scales are constructed with the Escota operating domain experts. Then, it is explained how to build a WAM aggregation operator w.r.t. each operating domain, in order to be consistent with the identified scales. The MACBETH method is the support of these first two steps. The problem related to the finite scales, that the experts use when assigning partial scores to an operation, is then considered. A method is proposed to ensure a logically sound interface between symbolic assessments and numerical computations in the framework of WAM aggregation. Then, a robustness analysis is proposed to determine the potential causes of overestimation or underestimation in the evaluation process of an operation.

2 CARDINAL SCALES OF EMERGENCY DEGREES

2.1 Nature of Scales

The purpose of this section is to explain how we have worked with Escota experts of the different operating domains in order to properly identify their emergency scales. There are one emergency scale for each criterion of the domain and one scale for the aggregated emergency value. In the following we will consider the case of the operating domain "carriageway". Eight criteria ($n=8$) are related to it: *security, durability, regulation, comfort, public image, environment protection, sanitary* and *social aspects*.

It has been checked a priori that Escota emergency scales are of cardinal nature: the emergency scale relative to any of the criteria is an interval scale.

Let us consider a finite set X. When the elements of X can be ranked w.r.t. to their attractiveness, this is ordinal information. It means that a number $n(x)$ can be associated to any element x of X such that:

$$\forall x, y \in X : [x \mathrm{P} y \Leftrightarrow n(x) \succ n(y)] \quad (1)$$

$$\forall x, y \in X : [x \mathrm{I} y \Leftrightarrow n(x) = n(y)] \quad (2)$$

where relation P « *is more attractive than* » is asymmetric and non transitive and relation I « *is as attractive as* » is an equivalence relation. *n(x)* defines an ordinal scale.

Based upon this first level of information, an interval scale can then be built. The next step consists in evaluating the difference of intensity of preference between elements of X. It implies the following constraints:

$$n(x) - n(y) = k\alpha, k \in \mathrm{N} \quad (3)$$

where k characterizes the intensity of preference and α enables to respect the limits of the domain (for example [0,1]). The resolution of a system of equations of type (1), (2) and (3) provides an interval scale. That's the principle used in the MACBETH method (Bana e Costa, 1994).

2.2 Emergency Scales and MACBETH Method

The problem of commensurability of the dimensions to be aggregated is at the heart of the MACBETH method. Aggregation can be envisaged only if the scales relative to the emergency criteria are commensurable (Clivillé, 2004). Then, MACBETH guarantees the consistency between the resulting partial scales and the WAM aggregation (Bana e Costa, 1994).

First, a training set of operations is constituted. A ranking of the operations in terms of emergency is established w.r.t. each criterion. At this stage, information is purely ordinal. Then, for each criterion, the solutions are compared pair to pair. Two fictive alternatives are introduced in the comparison process; they provide the reference values corresponding to the two emergency degrees: zero and one. The zero (resp. one) emergency degree corresponds to the threshold value under which operations are considered as not urgent at all (resp. highly urgent). The comparison then consists in quantifying the difference of emergency degree for each criterion. This difference is expressed in a finite set of labels: for example, "equivalent", "weak", "strong" and "extreme". The resulting set of constraints defines a linear programming problem. The solution of this problem provides the cardinal scale of emergency associated to one criterion. This step is repeated for each criterion.

Figure 1 illustrates this process for criterion *security*. The carriageway expert compares 10 operations {A… J} pair to pair. The real names of operations are not given for confidentiality reasons. Two fictive operations *urgent* (highly urgent) and *peu_urgent* (not urgent at all) complete the training base. The "positive" label in figure 1 introduces a more flexible constraint because it simply replaces any label with a higher degree than weak. The resulting cardinal scale is given at the right side of Figure 1.

Finally, this procedure is then applied to identify the weights of the WAM operator. The pair to pair comparison is carried out over the eight criteria of the carriageway domain (Figure 2). The resulting interval scale of weights is given in Figure 2. Let us note the weights p_i, $i = 1..n$ (n=8 for the carriageway domain). At this stage of the modelling, the carriageway expert has identified his 8 emergency scales and his WAM parameters. He is supposed to be able to compute the global degree of emergency of any operation when partial quotations u_i are available, w.r.t. each criterion:

$$WAM(OP) = \sum_{i=1}^{n} p_i u_i$$

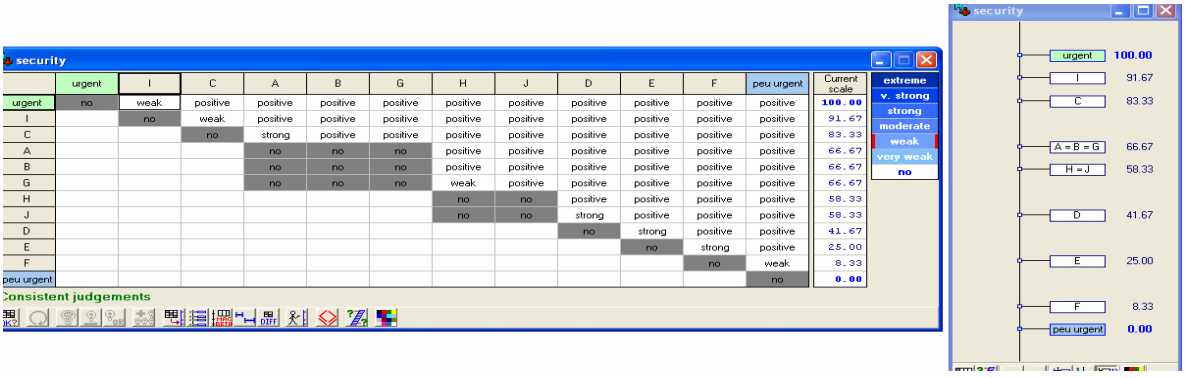


Figure 1: MACBETH - Pair to pair comparison of operations and cardinal scale for security criterion.

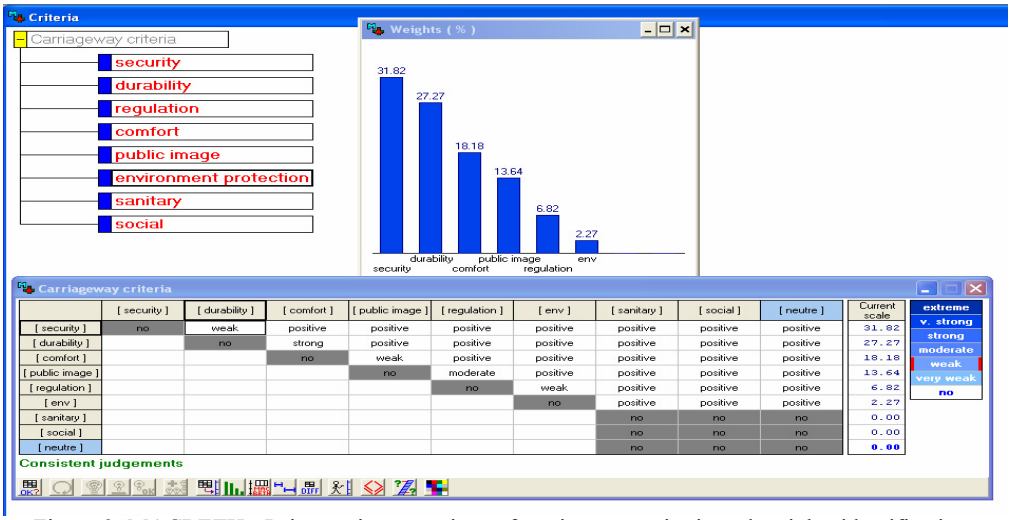


Figure 2: MACBETH - Pair to pair comparison of carriageway criteria and weights identification.

3 DISCRETE CARDINAL SCALES OF EMERGENCY

Partial scores aggregation does not cause any problem when quotations referred to continuous cardinal scales. As explained in section 1, it is more questionable when partial scores are expressed on a discrete or finite scale. Indeed, Escota experts express their assessment w.r.t. each criterion on a finite set of 3 labels $\{U_1, U_2, U_3\}$. The different U_i define a discrete cardinal scale. However, computing the WAM value necessitates assigning numerical values to each U_i. In the following, we describe the way this assignment can be achieved in a consistent manner with previous MACBETH identification phases.

A continuous cardinal scale has been identified with MACBETH method for the emergency scale of each criterion. The problem is now to assign a set of numerical values $\{u_1^i, u_2^i, u_3^i\}$ to $\{U_1, U_2, U_3\}$ for criterion *i*. Let us suppose the continuous cardinal scale for criterion *i* has been identified with a training set of *q* operations. These operations are grouped into 3 clusters corresponding to U_1, U_2, U_3. The computation of the clusters and their associated centres is achieved by minimizing the quadratic difference $\sum_{k=1}^{3}\sum_{j=1}^{q_k}(u_k^i - u^i(OP_j))^2$ where q_k is the number of operations in class U_k ($\sum_{k=1}^{3} q_k = q$) and $u^i(OP_j)$, *j=1..q*, the emergency degree of an operation OP_j computed with MACBETH (Figure 1).

In the example of Figure 1, the computation of clusters gives: $u_1^{security} = 0.91$, $u_2^{security} = 0.52$ and $u_3^{security} = 0.11$.

This assignment is repeated for each criterion relative to the carriageway domain. Then, the WAM can be numerically computed:

- For each criterion *i*, $i = 1..n$ ($n = 8$), a value U_k is affected to an operation *OP*. Let us note this emergency degree $U_{k(i)}$;
- *OP* is thus described by its vector of emergency degrees $[U_{k(1)}, .., U_{k(n)}]$;
- The corresponding vector of numerical values is: $\{u^1_{k(1)}, u^2_{k(2)}, .., u^n_{k(n)}\}$;

$$WAM(OP) = \sum_{i=1}^{n} p_i . u^i_{k(i)} \qquad (4)$$

The last constraint to be satisfied is that the *WAM* values must be converted in return into the semantic universe $\{U_1, U_2, U_3\}$. The output of the *WAM* operator must be discretized in $\{U_1, U_2, U_3\}$. The problem is thus to determine the centres of the U_k clusters of the aggregated emergency scale (*WAM* values).

Let us note that the *WAM* operator is idempotent. Therefore, we must have:

$$\forall U_k, k \in \{1,2,3\}, WAM(U_k, ..., U_k) = U_k \qquad (5)$$

A sufficient condition for (5) is that the centres of the U_k clusters of the aggregated emergency scale are the images of the corresponding U_k centres of the partial emergency scales by the *WAM* function, i.e.:

$$WAM(u_k^1,..,u_k^n) = \sum_{i=1}^{n} p_i.u_k^i = u_k^{Ag} \qquad (6)$$

where u_k^{Ag} is the centre of class U_k in the aggregated emergency scale.

Consequently, when an operation is defined by its partial emergency vector $[U_{k(1)},..,U_{k(n)}]$, equation (4) provides the numerical value

$$WAM(OP) = \sum_{i=1}^{8} p_i.u_{k(i)}^i \qquad (7)$$

Then, the attribution of a class U_k in the aggregated emergency scale is obtained through the following calculation:

$$\min_k \left| u_k^{Ag} - \sum_{i=1}^{n} p_i.u_{k(i)}^i \right| \qquad (8)$$

The value of k in $\{1,2,3\}$ that minimizes the expression in (8) provides the class U_k of operation *OP*.

Figure 3 summarizes the whole evaluation process of an operation *OP*. The validation of this process has been carried out with a test base of 23 operations in the carriageway domain. The carriageway expert has analysed each of these operations. For each of them, he has attributed emergency degrees in the Escota normalized semantic universe $\{U_1,U_2,U_3\}$ w.r.t. every of his 8 criteria.

Then, the aggregated emergency degree in this semantic universe can be computed using the 3-step process described in this paper (white arrows in Figure 3). Besides these computations, the expert has been asked to directly attribute an overall emergency degree to each of the 23 operations (grey arrow in Figure 3).

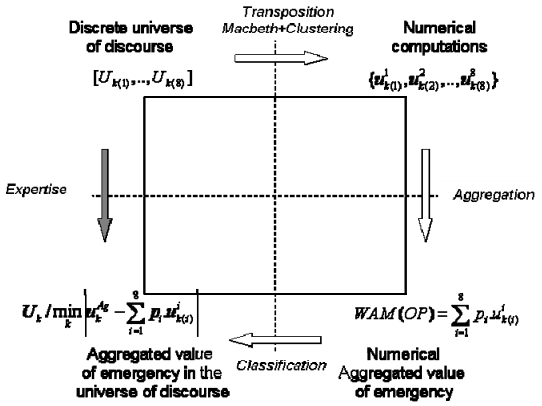


Figure 3: Evaluation process of an operation.

Figure 4 reports these data. The last line corresponds to the direct expert evaluation (grey arrow). The last but one line provides the corresponding computed values with the 3-step method (white arrows). No error has been observed. However, the poor semantic universe—only 3 labels—implied in our application can also partly explain such a perfect matching.

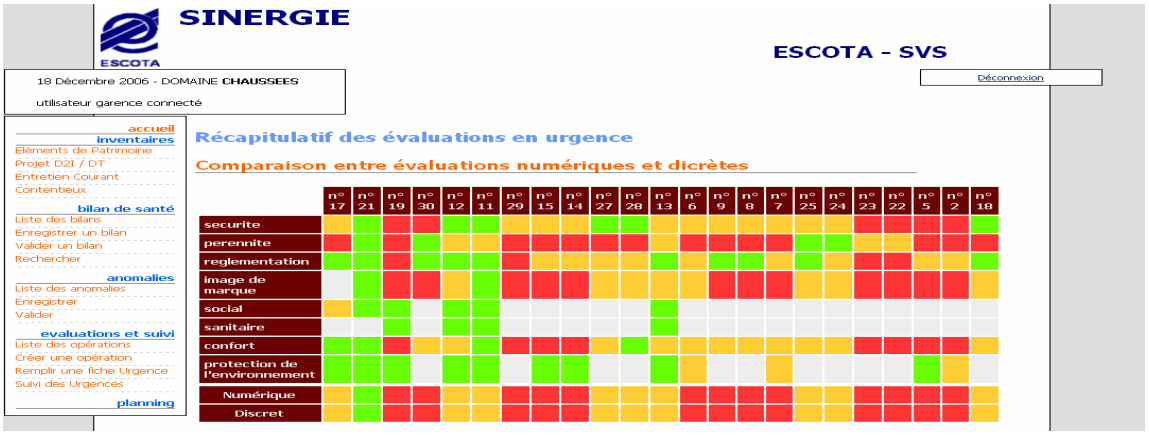


Figure 4: Tests on the evaluation method over a base of 23 operations

4 THE MC HIERARCHICAL EVALUATION BY ESCOTA

In this paper, the study was focused on the MC evaluation by the expert of an operating domain. However, as evocated in section 1, planning of operations, by Escota, is more complex. The emergency assessment by operating domain experts described here is only part of a hierarchical MC evaluation process. From symptoms detection on elements of infrastructure to operation planning, a similar MC evaluation is carried out at different functional levels in the Escota organization.

The complete information processing used for Escota preventive maintenance can be formalized as the following sequence of risk analysis. Periodic inspections are performed to detect and measure any malfunction symptoms as early as possible. The expert in charge of a domain then analyses these technical diagnoses and evaluates the situation seriousness. The official in charge of the operating network coordinates and ponders the experts' needs and demands. Each actor of this information processing system participates to a tripartite MC decision-making logic: measurement, evaluation and decision. To each step of this process corresponds a specific set of criteria and an aggregation operator: seriousness of a malfunction results from a prescribed aggregation of the symptoms quotation; the expert's interpretation of the diagnosis associates an emergency degree to the corresponding maintenance operation w.r.t. the criteria relating to his operating domain (technical risks assessment); finally, the manager attributes a priority degree to the operation on the basis of a set of more strategic criteria (strategic risks analysis).

This hierarchical MC evaluation process enables to breakdown the decision-making into elementary steps. Each step collaborates to the enrichment of information from measures to priority degrees and thus contributes to the final step, i.e. operation planning.

We have developed a dynamic Information Processing System (IPS) to support this hierarchical MC evaluation of the infrastructure condition and facilitate the way decision-makers use their reasoning capabilities through adequate information processing procedure.

Figure 5 illustrates the man machine-interface the expert has at his disposal to fulfil an emergency form relative to an operation. Finally, the emergency evaluation synthesis (Figure 6) can be consulted by the official in charge of the operation network before he proceeds to his own MC evaluation.

Figure 5: Keyboarding of an emergency form.

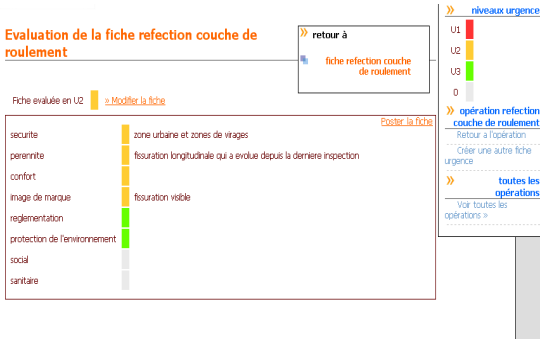


Figure 6: Emergency evaluation synthesis.

5 THE ROBUSTNESS ANALYSIS OF THE EVALUATION PROCESS

Let us now consider a last step in the evaluation process: assessment of the risk of erroneous estimation w.r.t. the emergency of an operation, i.e., the risk of underestimation or overestimation of the aggregated emergency score of an operation. It relies on a robustness analysis of the evaluation procedure based upon the WAM. Two aims are assigned to this step, it must answer the following questions: 1) when an erroneous partial estimation is done w.r.t. criterion *i*, what is the risk the aggregated emergency degree to be affected? 2) when an operation appears to be underestimated (resp. overestimated), which criteria could most likely explain this faulty result? The first question corresponds to *an a priori risk estimation* of erroneous evaluation; the second question is related to *a diagnosis analysis*.

Let us first define the notion of neighbourhood of a vector of emergency degrees $[U_{k(1)},..,U_{k(n)}]$ associated to an operation *OP*. The vectors of the neighbourhood of $[U_{k(1)},..,U_{k(n)}]$ are all the vectors $[U'_{k(1)},..,U'_{k(n)}]$ such that: $\forall i \in \{1..n\}, U'_{k(i)} = U_{k(i)}$ or $U'_{k(i)}$ is the value just above (resp. below) $U_{k(i)}$ (when

defined; indeed, there is no value below zero and no value above U_1). The neighbourhood is a set of vectors denoted $\mathrm{N}([U_{k(1)},..,U_{k(n)}])$. In the example in dimension 2 in Figure 8, $U_{k(1)} = U_2$ and $U_{k(2)} = U_2$. The values of component i ($i = 1 or 2$) of a neighbour vector may be U_2, U_1 or U_3. There are 8 neighbours. In the general case, the maximal number of neighbours is $3^n - 1$.

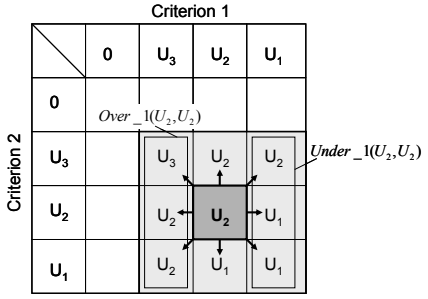


Figure 7: Neighbourhood of the vector of emergency degrees (U_2, U_2) in dimension 2.

5.1 Risk of Erroneous Estimation

The risk of misclassification of an operation due to an overestimation (resp. underestimation) w.r.t. a criterion *i* enables the expert in charge of a domain to assess the impact of an evaluation error w.r.t. criterion *i* on the overall emergency degree of the operation. The higher, the more carefully the partial appraisal w.r.t. criterion *i* must be carried out.

Compute $WAM(U)$

For each criterion *i* :

- Find all the vectors $U' = [U'_{k(1)},..,U'_{k(n)}]$ in $\mathrm{N}(U)$ such that $U'_{k(i)}$ takes the value just above $U_{k(i)}$ (when defined, else $U_{k(i)} = U_1$ and there is no risk of underestimation w.r.t. criterion *i* in this case). Note this set: $Under\_i(U)$
- Count the numbers of vectors U' in $Under\_i(U)$ such that $WAM(U')$ is higher than $WAM(U)$. Note this number n_{under_i}
- The risk of underestimation induced by criterion *i* for an operation characterized by U is then:

$$risk\_under(i) = \frac{n_{under\_i}}{|Under\_i(U)|}$$

The lower, the weaker the impact of the criterion to the global emergency degree. The risk analysis is based upon the following algorithm. We'll first consider the risk of underestimation for sake of simplicity. We consider that a value $U_{k(i)}$ is underestimated (resp. overestimated) when it should take the value just above $U_{k(i)}$ (resp. just below $U_{k(i)}$). This assumption means that the worst appraisal error w.r.t. one criterion can only correspond to the value just below or just above for this criterion.

Let's consider a vector $U = [U_{k(1)},..,U_{k(n)}]$

In the example in Figure 8, let us consider an assumption of underestimation w.r.t. criterion 1. The set $Under\_1(U_2,U_2)$ is represented in the figure. $|Under\_1(U_2,U_2)| = 3$; only (U_1,U_2) and (U_1,U_1) lead to an overall underestimation (the operation is evaluated U_2 whereas it should be U_1). Then, $n_{under\_1} = 2$ and $risk\_under(1) = 2/3$. It means that an underestimation w.r.t. criterion 1 for an operation characterized by (U_2,U_2) leads to an underestimation of the overall degree of emergency of the operation in 66% of the cases.

The algorithm is the same for the risk of overestimation. Nevertheless, in this case, when $U_{k(i)} = 0$, the risk of overestimation w.r.t. criterion *i* is null. Figure 9 and Figure 10 provide the results for the risk analysis when underestimation (Figure 9) and when overestimation (Figure 10) for all the vectors in Figure 4.

5.2 Diagnosis Analysis

When the degree of emergency of an operation is suspected to be overestimated (resp. underestimated), the diagnosis analysis consists in determining the most likely causes, i.e., the criteria that the most frequently entail an overestimation (resp. underestimation) of the operation when they're overestimated (resp. underestimated) themselves. The possibility that criterion *i* is a cause of overestimation (resp. underestimation) assuming an overestimation (resp. underestimation) of the overall emergency degree of the operation— is computed in the diagnosis step.

Let us consider the algorithm in case of underestimation (resp. overestimation).

$Diag\_under(i)$ gives the rate that an underestimation w.r.t. criterion *i* be a potential cause of underestimation of the overall emergency degree of an operation (idem for overestimation).

Let's consider a vector $U = [U_{k(1)},..,U_{k(n)}]$

Compute $WAM(U)$

Compute $\mathrm{N}(U)$ and its cardinal $|\mathrm{N}(U)|$

- Compute $WAM(U')$ for each $U' = [U'_{k(1)},..,U'_{k(n)}]$ in $\mathrm{N}(U)$
- Let us note $Higher\_\mathrm{N}(U)$ (resp. $Lower\_\mathrm{N}(U)$), the set of vectors U' in $\mathrm{N}(U)$ such that $WAM(U') > WAM(U)$ (resp. $WAM(U') < WAM(U)$)
- For each criterion i, count the number n'_{under_i} (resp. n'_{over_i}) of times criterion i is underestimated (resp. overestimated) in a vector of $Higher\_\mathrm{N}(U)$ (resp. $Lower\_\mathrm{N}(U)$), i.e., $U'_{k(i)}$ takes the value just above $U_{k(i)}$ (resp. just below $U_{k(i)}$) in $Higher\_\mathrm{N}(U)$ (resp. $Lower\_\mathrm{N}(U)$)

Compute for each criterion i:

$$Diag\_under(i) = \frac{n'_{under\_i}}{|Higher\_\mathrm{N}(U)|}$$

(resp. $Diag\_over(i) = \frac{n'_{over\_i}}{|Lower\_\mathrm{N}(U)|}$)

Figure 11 concerns underestimation diagnosis and Figure 12 overestimation diagnosis for the base of operations in Figure 4. A rate indicates the possibility a criterion is underestimated itself (resp. overestimated) when the overall emergency degree of the concerned operation is underestimated (resp. overestimated).

6 CONCLUSION

In civil engineering, the culture of numbers is strongly developed. People commonly manipulate symbolic labels but attribute them numerical values when necessary without further care. A typical case is when aggregation procedures are required. We have proposed a methodology that enables 1) experts to express their judgement values in their own discrete semantic universe, 2) to convert the labels in adequate numerical values using the MACBETH method and clustering techniques, 3) to compute the WAM based aggregated value and convert it in return into the experts' semantic universe 4) to carry out a robustness analysis of the evaluation process to assess the risk of misclassification of the operations and to diagnose these misclassifications. This method is implemented in an IPS—SINERGIE—that supports decisions concerning maintenance operations planning by the motorway operator Escota.

REFERENCES

Saaty, T.L. 1980. The Analytic Hierarchy Process. *McGraw-Hill ,* New York.

Bana e Costa, C.A., Vansnick, J.C., 1994. MACBETH - an interactive path towards the construction of cardinal value functions. *International transactions in Operational Research*, *1, 489-500.*

Clivillé, V., 2004. Approche Systémique et méthode multicritère pour la définition d'un système d'indicateurs de performance. *Thèse de l'Université de Savoie,* Annecy.

Jullien, S., Mauris, G., Valet, L., Bolon, Ph., 2006. Decision aiding tools for Animated film selection from a mean aggregation of criteria preferences over a finite scale. *11th Int. Conference on Information processing and Management of uncertainty in Knowledge-Based Systems,* Paris, France.

Grabisch, M., 2006. Representation of preferences over a finite scale by a mean operator. *To appear in mathematical Social Sciences,* downloadable at www-sysdef.lip6.fr/~grabisch.

	17	21	19	30	12	11	29	15	14	27	28	13	6	9	8	7	25	24	23	22	5	2	18
env	11.0%	44.0%	0%	0%	15.0%	2.0%	0%	0%	0%	26.0%	13.0%	41.0%	0%	0%	0%	0%	18.0%	21.0%	0%	0%	0%	0%	21.0%
sanitary	11.0%	42.0%	0%	0%	14.0%	1.0%	0%	0%	0%	26.0%	13.0%	39.0%	0%	0%	0%	0%	18.0%	21.0%	0%	0%	0%	0%	21.0%
comfort	25.0%	58.0%	0%	0%	25.0%	4.0%	0%	0%	0%	40.0%	27.0%	55.0%	0%	0%	0%	0%	28.0%	32.0%	0%	0%	0%	0%	37.0%
regulation	11.0%	45.0%	0%	0%	16.0%	2.0%	0%	0%	0%	34.0%	20.0%	43.0%	0%	0%	0%	0%	20.0%	25.0%	0%	0%	0%	0%	23.0%
security	32.0%	67.0%	0%	0%	37.0%	4.0%	0%	0%	0%	62.0%	35.0%	76.0%	0%	0%	0%	0%	41.0%	46.0%	0%	0%	0%	0%	55.0%
durability	15.0%	73.0%	0%	0%	28.0%	4.0%	0%	0%	0%	32.0%	17.0%	62.0%	0%	0%	0%	0%	39.0%	44.0%	0%	0%	0%	0%	27.0%
social	11.0%	42.0%	0%	0%	14.0%	1.0%	0%	0%	0%	26.0%	13.0%	39.0%	0%	0%	0%	0%	18.0%	21.0%	0%	0%	0%	0%	21.0%
public image	18.0%	56.0%	0%	0%	26.0%	4.0%	0%	0%	0%	44.0%	27.0%	58.0%	0%	0%	0%	0%	29.0%	34.0%	0%	0%	0%	0%	39.0%

	17	21	19	30	12	11	29	15	14	27	28	13	6	9	8	7	25	24	23	22	5	2	18
env	2.0%	3.0%	0.0%	53.0%	7.0%	30.0%	12.0%	17.0%	17.0%	1.0%	6.0%	0.0%	41.0%	34.0%	34.0%	28.0%	13.0%	11.0%	6.0%	6.0%	0.0%	0.0%	1.0%
sanitary	2.0%	3.0%	0.0%	53.0%	6.0%	29.0%	12.0%	16.0%	16.0%	1.0%	6.0%	0.0%	34.0%	34.0%	34.0%	22.0%	13.0%	11.0%	6.0%	6.0%	0.0%	0.0%	1.0%
comfort	8.0%	7.0%	1.0%	69.0%	13.0%	46.0%	19.0%	24.0%	24.0%	3.0%	14.0%	2.0%	51.0%	51.0%	51.0%	39.0%	20.0%	18.0%	12.0%	12.0%	2.0%	1.0%	3.0%
regulation	3.0%	3.0%	1.0%	58.0%	9.0%	32.0%	19.0%	23.0%	23.0%	1.0%	6.0%	1.0%	40.0%	35.0%	35.0%	28.0%	14.0%	14.0%	7.0%	7.0%	1.0%	0.0%	1.0%
security	8.0%	9.0%	1.0%	72.0%	18.0%	63.0%	35.0%	47.0%	47.0%	3.0%	18.0%	2.0%	74.0%	80.0%	80.0%	56.0%	30.0%	27.0%	19.0%	19.0%	2.0%	1.0%	3.0%
durability	8.0%	9.0%	1.0%	96.0%	17.0%	49.0%	24.0%	30.0%	30.0%	3.0%	11.0%	2.0%	48.0%	48.0%	48.0%	36.0%	39.0%	34.0%	18.0%	18.0%	2.0%	1.0%	3.0%
social	2.0%	3.0%	0.0%	53.0%	6.0%	29.0%	12.0%	16.0%	16.0%	1.0%	6.0%	0.0%	34.0%	34.0%	34.0%	22.0%	13.0%	11.0%	6.0%	6.0%	0.0%	0.0%	1.0%
public image	3.0%	5.0%	1.0%	60.0%	15.0%	44.0%	24.0%	29.0%	29.0%	3.0%	14.0%	2.0%	53.0%	48.0%	48.0%	33.0%	20.0%	18.0%	12.0%	12.0%	2.0%	1.0%	3.0%

Figures 8 and 9: risk of overall underestimation (resp. overestimation) of the operations induced by partial underestimations (resp. overestimation) w.r.t. criteria.

	17	21	19	30	12	11	29	15	14	27	28	13	6	9	8	7	25	24	23	22	5	2	18
durability	45%	57%	0%	0%	67%	100%	0%	0%	0%	40%	42%	52%	0%	0%	0%	0%	71%	67%	0%	0%	0%	0%	41%
security	97%	53%	0%	0%	86%	100%	0%	0%	0%	79%	87%	64%	0%	0%	0%	0%	75%	71%	0%	0%	0%	0%	84%
comfort	77%	45%	0%	0%	59%	90%	0%	0%	0%	51%	66%	46%	0%	0%	0%	0%	51%	49%	0%	0%	0%	0%	56%
public image	55%	44%	0%	0%	62%	90%	0%	0%	0%	56%	66%	48%	0%	0%	0%	0%	53%	52%	0%	0%	0%	0%	60%
env	35%	35%	0%	0%	36%	45%	0%	0%	0%	33%	33%	35%	0%	0%	0%	0%	33%	33%	0%	0%	0%	0%	33%
regulation	35%	35%	0%	0%	37%	45%	0%	0%	0%	43%	51%	36%	0%	0%	0%	0%	37%	39%	0%	0%	0%	0%	35%
social	33%	33%	0%	0%	33%	33%	0%	0%	0%	33%	33%	33%	0%	0%	0%	0%	33%	33%	0%	0%	0%	0%	33%
sanitary	33%	33%	0%	0%	33%	33%	0%	0%	0%	33%	33%	33%	0%	0%	0%	0%	33%	33%	0%	0%	0%	0%	33%

	17	21	19	30	12	11	29	15	14	27	28	13	6	9	8	7	25	24	23	22	5	2	18
durability	100%	100%	100%	60%	82%	57%	66%	61%	61%	100%	60%	100%	46%	46%	46%	53%	94%	96%	93%	93%	100%	100%	100%
security	100%	100%	100%	46%	90%	72%	96%	95%	95%	100%	100%	100%	71%	78%	78%	84%	73%	75%	100%	100%	100%	100%	100%
comfort	100%	78%	100%	44%	66%	53%	53%	48%	48%	100%	80%	100%	50%	50%	50%	57%	50%	51%	62%	62%	100%	100%	100%
public image	45%	60%	100%	38%	74%	50%	66%	58%	58%	100%	80%	100%	51%	46%	46%	50%	50%	51%	62%	62%	100%	100%	100%
env	35%	34%	33%	33%	35%	34%	33%	35%	35%	33%	33%	40%	40%	33%	33%	42%	33%	33%	33%	33%	33%	50%	33%
regulation	40%	39%	100%	35%	43%	36%	53%	46%	46%	33%	33%	60%	39%	34%	34%	42%	35%	41%	37%	37%	50%	50%	33%
social	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%
sanitary	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%	33%

Figures 10 and 11: rates of causes of underestimation (resp. overestimation) diagnoses.

COMPARYING A TABU SEARCH PROCESS
Using and Not Using and Intensification Strategy to Solve the Vehicle Routing Problem

Etiene Pozzobom Lazzeris Simas
Universidade do Vale do Rio dos Sinos. Av. Unisinos, 950, São Leopoldo, Brasil
Etiene.simas@terra.com.br

Arthur Tórgo Gómez
Universidade do Vale do Rio dos Sinos. Av. Unisinos, 950, São Leopoldo, Brasil
breno@unisinos.br

Keywords: Vehicle Routing Problem; Tabu Search, Intensification Strategy.

Abstract: In this paper we propose a Tabu Search algorithm to solve the Vehicle Routing Problem. The Vehicle Routing Problem are usually defined as the problem that concerns in creation of least cost routs to serve a set of clients by a fleet of vehicles. We develop an intensifications strategy to diversify the neighbours generated and to increase the neighbourhood size. We had done experiments using and not using the intensification strategy to compare the performance of the search. The experiments we had done showed that an intensification strategy allows an increase on the solutions quality.

1 INTRODUCTION

The Vehicle Routing Problem (VRP) is a NP-Hard problem (Lenstra and Rinooy Kan, 1981) that is usually dealt within the logistic context (Ho and Haugland, 2004; Xu and Kelly, 1996). It can be described as a set of customers that have to be served by a fleet of vehicles, satisfying some constraints (Laporte, 1992; Xu and Kelly, 1996). The Transport is one of the most costly activities in logistic, typically varying in one or two thirds of the total costs (Ballou, 2001). Therefore, the necessity of improving the efficiency of this activity has great importance. A small percentage saved with this activity could result in a substantial saving total (Bodin, Golden and Assad, 1983). There many variants and constraints that can be considered, i.e. it can be considered that the fleet may be heterogeneous the vehicles must execute collections and deliveries, there may exist more than one depot, etc. In this paper we are dealing with the classic version of this problem, were just the vehicle capacity constraint are considered.

2 THE VEHICLE ROUTING PROBLEM

A classical definition is presented in Barbarasolgu and Ozgur (Barbarasolgu and Ozgur, 1999). The VRP is defined in a complete, undirected graph G=(V.A) where a fleet of Nv vehicle of homogeneous capacity is located. All remaining vertices are customers to be served. A non-negative matrix $C=(c_{ij})$ is defined on A representing the distance between the vertices. The costs are the same in both directions. A non-negative demand di, is associated with each vertex representing the customer demand at vi. The routes must start and finish at the depot. The clients must be visited just once, by only one vehicle and the total demand of the route can't exceed the capacity Qv of the vehicle. In some cases, there is a limitation on the total route duration. In this case, t_{ij} is defined to represent travel time for each (vi,vj), ti represents the service time at any vertex vi and is required that the total time duration of any route should not exceed Tv. A typical formulation based on Barbarasoglu and Ozgur ones (Barbarasolgu and Ozgur, 1999) are used in this paper:

$$\text{Minimize} \sum_i \sum_j \sum_v c_{ij} X^v_{ij} \quad (1)$$

Subject:

$$\sum_i \sum_v X^v_{ij} = 1 \text{ for all j} \quad (2)$$

$$\sum_j \sum_v X^v_{ij} = 1 \text{ for all i} \quad (3)$$

$$\sum_i X^v_{ip} - \sum_j X^v_{pj} = 0 \text{ for all p,v} \quad (4)$$

$$\sum_i d_i \left(\sum_j X^v_{ij} \right) <= Q_v \text{ for all v} \quad (5)$$

$$\sum_{j=1}^{n} X^v_{0j} <= 1 \text{ for all v} \quad (6)$$

$$\sum_{i=1}^{n} X^v_{i0} <= 1 \text{ for all v} \quad (7)$$

$$X^v_{ij} \in Z \text{ for all i,j and v} \quad (8)$$

Where X^v_{ij} are binary variables indicating if arc(vi,vj) is traversed by vehicle v. The objective function of distance/cost/time is expressed by eq. (1). Constraints in eqs (2) and (3) together state that each demand vertex is served by exactly one vehicle. The eq. (4) guarantees that a vehicle leaves the demand vertex as soon as it has served the vertex. Vehicle capacity is expressed by (5) where Q_v is the capacity. Constraints (6) and (7) express that vehicle availability can't be exceeded. The sub tour elimination constraints are given in eq.(8) where Z can be defined by:

$$Z = \left\{ \left(X^v_{ij} \right) : \sum_{i \in B} \sum_{j \in B} X^v_{ij} <= |B| - 1 \right. \quad (8)$$

$$\left. \text{for } B \subseteq V / \{0\}; |B| >= 2 \right\}$$

3 RESOLUTIONS METHODS

Since VRP is Np-Hard to obtain good solutions in an acceptable time, heuristics are used and this is the reason why the majority of researchers and scientists direct their efforts in heuristics development (Thangiah and Petrovik, 1997; Nelson et al, 1985; Xu and Kelly, 1996). Osman and Laporte (Osman and Laporte, 1996) define heuristic as a technique, which seeks good solutions at a reasonable computational cost without being able to guarantee the optimality. Laporte et al (Laporte et al, 2000) define two main groups of heuristics: classical heuristics, developed mostly between 1960 and 1990, and metaheuristics. The classical heuristics are divided in three groups: constructor methods, two-phase methods and improvement methods. Since 1990, the metaheuristics have been applied to the VRP problem. To Osman and Laporte (Osman and Laporte, 1996) a metaheuristic is formally defined as an iterative generation process which guides a subordinate heuristic by combining intelligently different concepts for exploring and exploiting the search space in order to find efficiently near-optimal solutions. Several metaheuristics have been proposed to solve the VRP problem. Among these ones, Tabu Search are considered the best metaheuristic for VRP. To review some works with Tabu Search and others metaheuristics some readings are suggested (Cordeau et al, 2002; Tarantilis et al, 2005).

3.1 Tabu Search

It was proposed by Glover (Glover, 1989) and had its concepts detailed in Glover and Laguna (Glover and Laguna, 1997). It's a technique to solve optimization combinatorial problems (Glover, 1989) that consists in an iterative routine to construct neighborhoods emphasizing the prohibition of stopping in an optimum local. The process that Tabu Search searches for the best solution is through an aggressive exploration (Glover and Laguna, 1997), choosing the best movement for each iteration, not depending on if this movement improves or not the value of the actual solution. In Tabu Search development, intensification and diversification strategies are alternated through the tabu attributes analysis. Diversification strategies direct the search to new regions, aiming to reach whole search space while the intensification strategies reinforce the search in the neighborhood of a solution historically good (Glover and Laguna, 1997). The stop criterion makes it possible to stop the search. It can be defined as the interaction where the best results were found or as the maximum number of iteration without an improvement in the value of the objective function. The tabu list is a structure that keeps some solution's attributes that are considered tabu. The objective of this list is to forbid the use of some solutions during some defined time.

4 VRP APPLICATION

The application developed are divided into three modules:

a) Net Generation module: This module generates the nets that will be used in the application using vertices coordinates and demands given.
b) Initial Solution module: This module generates the initial solutions of the nets. The initial solutions are created thought the use of an algorithm implementing the Nearest Insertion heuristic (Tyagi, 1968; Cook et al, 1998).
c) Tabu Search module: This module performs the tabu search algorithm. The Tabu Search elements that were used are now detailed. The stop criterion adopted is the maximum number of iterations without any improvement in the value of the objective value. The tabu list keeps all the routes and the cost of the solution, forbidden these routes to be used together during the tabu tenure defined. And elite solution list is used to keep the best results that were found during the search. It was proposed an intensification strategy to be used every time when the search executes 15 iterations without an improvement in the objective function value. In this strategy we visit every solution that is in elite list generating a big neighbourhood for each one. There were defined two movements to neighbourhood generation. V1, that makes the exchange of vertices and V2, that makes the relocation of vertices. In V1, one route r1 is selected and than one vertex of this route is chosen. We try to exchange this vertex with every vertex of all the other routes. The exchange is done if the addition of the two new demands doesn't exceed the vehicle's capacity of both routes. This procedure is done for every vertex of the route r1. To every exchange that is made, one neighbour is generated. In V2, we select one route and choose one vertex and then we try to reallocate it into all others routes, if it doesn't exceed the vehicle capacity of the route. When a vertex can be insert into a route, we try to insert it into all possible positions inside this route. To every position that a vertex is inserted, one neighbour is generated.
When these movements are used in the search with intensification, they are called V1' and V2' because with intensification, not only one route is selected like in V1 and V2, but also all routes of the solution are chosen. Aiming increase the neighbourhood size and the diversification between the solutions, we proposed to use the movements alone and together.

5 COMPUTACIONAL EXPERIENCE

The computational experiments were conducted on problems 1, 2, 3, 4 and 5 of Christofides Mingozzi and Toth (Christofides Mingozzi and Toth, 1979). These problems contain 50, 75, 100, 150 and 199 vertices and one depot respectively and they are frequently used in papers for tests purposes. The objective of the experiments was to compare the search process using and not using intensification strategy using the different movements proposed. There were proposed 9 values to Nbmax {100, 250, 500, 750, 1000, 1250, 1500, 1750, and 2000} and 6 values to Tabu List size {10, 25, 50, 75, 100, 200}. For every problem, the experiments were divided into 6 groups according with the used movements. Table 1 shows these groups.

Table 1: Groups of experiments divided by movements.

Search mode	Used movement
Using Intensification	V1
Using Intensification	V2
Using Intensification	V1,V2
Not Using Intensification	V1 + V1'
Not Using Intensification	V2 + V2'
Not Using Intensification	V1V2 + V1' V2'

There were generated 54 experiments for each group combining all values proposed to Nbmax with all Tabu List size. So, for each problem there were generated 162 experiments using intensification and 162 experiments not using it. Two types of analyses were done. In one type it was evaluated the best result obtained for a fixed value of Nbmax used with all Tabu List size and in other type it was evaluated the best results obtained for a fixed size of Tabu List used with all values proposed to Nbmax. Analyses had also been made comparing the best result found for each group of experiment, in this case comparing the quality of the different movements.

5.1 Analysing the Nbmax Variation for Each Tabu List Size

By analysing the results in this perspective it will be evaluate the variation of the Nbmax for each Tabu List size. The objective is verified if big values of Nbmax can improved the quality of Tabu Search process. We create a "lower average" for the average from results obtained with Nbmax = 100 and Nbmax = 250 and an "upper average" for the average from results obtained with Nbmax = 1750 and 2000. For all problems, analysing each one of the 6 groups of experiments done, the "upper average" were always lesser than the "lower average", indicating that big values of Nbmax can improve the search quality. Figure 1 shows an example of the graphics

generated with the results of the search process. It's clear to see that an increase in Nbmax value can improve the search quality, by decreasing the results costs of the solutions.

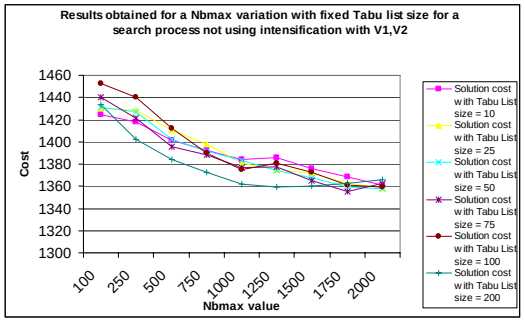


Figure 1: Costs obtained from Nbmax variation with different Tabu List size for problem 5 using V1 and V2 without intensification.

Tables 2 to 6 show the number of best results that were found in each Nbmax value:

Table 2: Quantity and localization of the best Results found for problem 1.

	500	750	1000	1250	1500	1750	2000
Int.	1	1	3	3	3	4	3
No Int.	0	0	1	2	4	6	5

Table 3: Quantity and localization of the best Results found for problem 2.

	500	750	1000	1250	1500	1750	2000
Int.	1	0	1	3	4	2	7
No Int.	0	0	1	2	3	6	6

Table 4: Quantity and localization of the best Results found for problem 3.

	500	750	1000	1250	1500	1750	2000
Int.	0	1	0	2	1	4	10
No Int.	0	1	0	1	2	1	12

Table 5: Quantity and localization of the best Results found for problem 4.

	500	750	1000	1250	1500	1750	2000
Int.	0	0	1	2	3	1	11
No Int.	0	0	1	4	1	2	10

Table 6: Quantity and localization of the best Results found for problem 5.

	500	750	1000	1250	1500	1750	2000
Int.	0	0	2	3	0	1	12
No Int.	0	0	0	1	1	5	11

These tables shown that most best results were found when the search used big values of Nbmax.

5.2 Analysing the Tabu List Size Variation for Each Nbmax Value

By analysing the results in this perspective it will be evaluate the variation of the tabu list size for each Nbmax value. The objective is verified if big Tabu list size can improved the quality of Tabu Search process. For problem 1, not using intensification, 66,66% of the results were found with Tabu list size >=75. Using intensification it was 48,14%. For problem 2 the percentage were 55,55% and 59,25%, not using and using intensification. For problem 3 the percentage were 77,77% and 96,29% not using and using intensification. For problem 4 these percentage were 74,05% and 77,77% and for problem 5 they were 63,88% and 92,59%. So by analysing these results we can see that big tabu list size can improve the quality of the search process.

5.3 Comparing the Search Process using and not using Intensification

This analysis intend to compare the search process using and not using the intensification strategy to see if it can improve the results generated.
Figure 2 shows an example of the graphics done with the results obtained in both search process to compare the quality of the different search process.

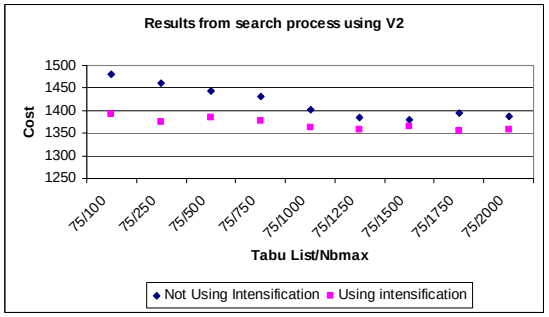


Figure 2: Costs obtained from both search process for Tabu List size = 75 and using V2 for problem 5.

This figure shows that an intensification strategy increase all results of the search process using V2 for problem 5. A comparison with the results generated by the search process using and not using intensification was done. Figures 3 to 7 shows the percentage of results that were improved with the intensification strategy. Figure 3 shows that for problem 1, from 162 results that were generated 97 were improved with intensification.

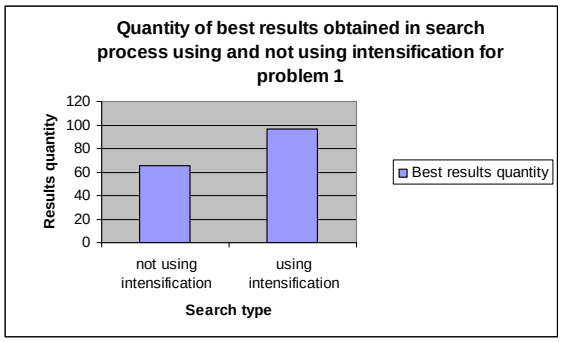


Figure 3: Improve caused by the intensification search for problem 1.

For problem 2, from 162 results, 121 were improved using intensification strategy.

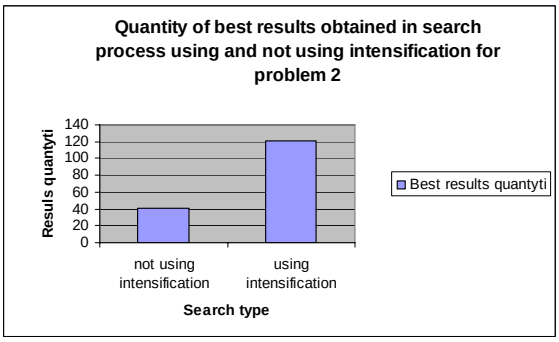


Figure 4: Improve caused by the intensification search for problem 1.

For problem 3, from 162 results 102 were improved using intensification strategy.

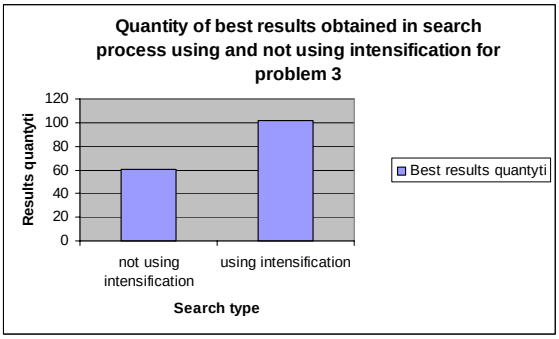


Figure 5: Improve caused by the intensification search for problem 3.

For problem 4, from 162 results 117 were improved using intensification strategy.

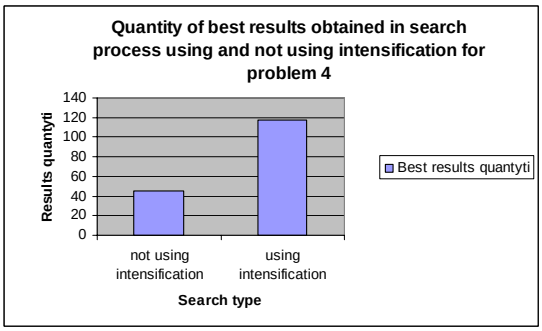


Figure 6: Improve caused by the intensification search for problem 4.

For problem 5, from 162 results 135 were improved using intensification strategy.

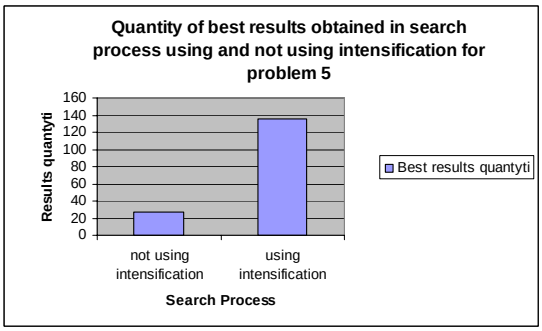


Figure 7: Improve caused by the intensification search for problem 5.

The figures show that an increase in solution quality of, at least, 50% happens when intensification strategy is used
The average results and the standard deviation are shown for problem 1, 2, 3, 4 and 5 in tables 7 to 11.

Table 7: Average results and standard deviation for problem 1.

Problem 1				
	Average		Standard Deviation	
	Not Using Intensive		Using Intensive	
V1	657,55	650,95	24,79	28,30
V2	582,08	570,30	21,16	22,21
V1,V2	537,36	542,01	8,56	13,62

Table 8: Average results and standard deviation for problem 2.

Problem 2				
	Average		Standard Deviation	
	Not Using Intensive.		Using Intensive	
V1	951,07	943,02	21,68	25,84
V2	895,75	883,03	20,67	21,71
V1,V2	867,96	863,06	13,00	14,70

Table 9: Average results and standard deviation for problem 3.

Problem 3				
	Average		Standard Deviation	
	Not Using Intensive.		Using Intensive	
V1	954,01	948,12	19,60	16,95
V2	903,63	901,17	14,50	17,84
V1,V2	879,90	870,10	15,05	20,73

Table 10: Average results and standard deviation for problem 4.

Problem 4				
	Average		Standard Deviation	
	Not Using Intensive.		Using Intensive	
V1	1215,35	1210,79	12,79	10,70
V2	1124,93	1118,83	15,66	12,52
V1,V2	1087,72	1079,54	18,15	16,70

Table 11: Average results and standard deviation for problem 5.

Problem 5				
	Average		Standard Deviation	
	Not Using Intensive.		Using Intensive	
V1	1569,11	1561,59	10,12	16,22
V2	1421,41	1393,56	34,40	12,75
V1,V2	1387,93	1377,82	26,64	16,37

From the results presented in tables 7 to 11 we can see that the results generated by the movements grouped are better than the results obtained using the movements alone. The reason for this is that when movements are used together the size of the neighbourhood generated is bigger than the neighbourhood generated by V1 or V2 alone. The movements together also cause an increase of the diversification of the solutions. And when the search generates more results, it is doing a deeper search in the space. Of course, as was shown, the intensification strategy helps the search to produce more qualified results.

When comparing the V1 and V2 movements, we can see that V2 produce results more qualified. If we analyse the policy behind the movement, we can say that V2 is more flexible than V1. V1 needs that two constraints are satisfied to generate one neighbour. While in V2, just one demand capacity must be verified (the capacity of the vehicle that serve the route where the vertex are being allocated) in V1, both routes must be verified to see if the vehicles capacities aren't exceeded.

Next table shows the best results obtained for each problem. All the best results were obtained during the search using movements V1 and V2 together and with the intensification strategy.

Table 12: Best results obtained.

Problem	Best Result
1	525,42
2	847,82
3	837,79
4	1061,07
5	1352,74

5.4 Comparisons

Aiming to evaluate the quality of the application developed, some papers were selected from the literature to compare the results. There were selected some classical heuristic and some papers that also used Tabu Search to solve the VRP. The paper selected were: {WL} Willard (1989), {PF}Pureza and França (1991), {OM1} Osman (1991), {OM2} Osman (1993), {RG} Rego (1998), {GHL} Gendreau, Hertz and Laporte (1994), {BO} Barbarasoglu and Ozgur (1999), {XK} Xu and Kelly (1996), {TV} Toth and Vigo (2003), {CW} Clarke and Wright (1964), {GM} Gillet and Miller (1974), {MJ} Mole and Jamenson (1976), {CMT} Christofides, Mingozzi and Toth (1979).

Table 13 shows the comparison done with the results from the papers. The results were obtained in Barbarasolgu and Ozgur (Barbarasolgu and Ozgur, 1999) and in Gendreau, Hertz and Laporte (Gendreau, Hertz and Laporte, 1994). In the first columns the paper used is shown. Columns 2 and 4 present the best results from the paper and columns 3 and 5 shows the difference in percentage from the results obtained in this paper to the paper compared. This difference was called "gap". The (+) indicate that our result is that percentage more than the result from the paper. The (-) indicate that our results is that percentage minor than the result from the paper.

Table 13: Best Results and gap for problem 1 and 2.

	Problem 1		Problem 2	
	Best	%Gap	Best	%Gap
WL	588	11,91(-)	893	5,33(-)
RG	557,86	6,17(-)	847	0,10(+)
PF	536	2,10(-)	842	0,69(+)
OM1	524,61	0,15(+)	844	0,45(+)
OM2	524,61	0,15(+)	844	0,45(+)
GHL	524,61	0,15(+)	835,77	1,42(+)
BO	524,61	0,15(+)	836,71	1,31(+)
XK	524,61	0,15(+)	835,26	1,48(+)
TV	524,61	0,15(+)	838,60	1,09(+)
CW	578,56	10,11(-)	888,04	4,74(-)
GM	546	3,92(-)	865	2,03(-)
MJ	575	9,44(-)	910	7,33(-)
CMT	534	1,63(-)	871	2,73(-)

Table 14: Best Results and gap for problem 3 and 4.

	Problem 3		Problem 4	
	Best	%Gap	Best	%Gap
WL	906	8,14(-)	-	-
RG	832,04	0,69(+)	1074,21	1,31(+)
PF	851	1,58(-)	1081	1,88(-)
OM1	835	0,33(+)	1052	0,85(+)
OM2	838	0,03(-)	1044,35	1,58(+)
GHL	829,45	1,00(+)	1036,16	2,35(+)
BO	828,72	1,08(+)	1043,89	1,62(+)
XK	826,14	1,39(+)	1029,56	2,97(+)
TV	828,56	1,10(+)	1028,42	3,08(+)
CW	878,70	4,88(-)	1204	13,47(-)
GM	862	2,89(-)	1079	1,69(-)
MJ	882	5,28(-)	1259	18,65(-)
CMT	851	1,58(-)	1093	3,01(-)

Table 15: Best Results and gap for problem 5.

	Problem 5	
	Best	%Gap
WL	-	-
RG	1352,88	0,014(-)
PF	-	-
OM1	1354	0,09(-)
OM2	1334,55	1,34(+)
GHL	1322,65	2,22(+)
BO	1306,16	3,44(+)
XK	1298,58	4,00(+)
TV	1291,45	4,53(+)
CW	1540	13,84(-)
GM	1389	2,68(-)
MJ	1545	14,21(-)
CMT	1418	4,82(-)

By analysing these tables we can see that our application produce more qualified results than all the classical heuristics used in comparison because our result was better than all of the heuristic results. Comparing with other tabu search algorithm, we can say that our algorithm is very competitive. It dominates at least 2 results from the 9 used for each problem. Moreover, the results generated were less than 5% of the other results for all cases. And in 25 cases out of 45 this percentage is minor than 2%.

6 FINAL CONSIDERATIONS

In this paper it was proposed an application using Tabu Search to solve the vehicle routing problem. This application was divided into 3 modules: a net generation module, an initial solution module and tabu search module. We used two movements based in relocation of vertices and exchange of vertices to create the neighbourhood. We use the movements alone and together, intending to diversify the solutions. We used an elite list solution to keep the best results found during the search. We propose an intensification strategy to use every time the search executes 15 iterations without improvement in objective value. We proposed some experiments to test if the solution quality increase or not with the increase in Nbmax value and in Tabu List size. We also compare the search process using and not using intensifications intending to see if this solution´s quality is improved with the Intensification strategy. The experiments showed that big values to Nbmax and Tabu list size could improve the results. From the experiments we also can see that an intensification strategy can improve the quality of the search.

REFERENCES

Ballou, R.H. 2001 Gerenciamento da cadeia de Suprimentos – Planejamento, Organização e Logística Empresarial, 4Ed, Porto Alegre: Bookman

Barbarasoglu,G., Ozgur, D. 1999. “A tabu search algorithm for the vehicle routing problem”, Computers & Operations Research 26, 255-270

Bodin, L.D, Golden, B.L., Assad, A.A., Ball, M.O. 1983 “Routing and Scheduling of vehicles and crews: The State of the Art”. Computers and Operations Research 10, 69-211

Clarke, G, Wright, J.W. 1964 “Scheduling of vehicles from a central depot to a number of delivery points”. Operations Research 12: 568-581

Christofides, N.; Mingozzi, A.; Toth, P.1979. The Vehicle Routing Problem.In: Christofides, Nicos. Combinatorial Optimization UMI, 1979

Cook,W.J, Cunningham, W.H, Pulleyblank, W.R, Schrijver, A.1998. Combinatorial Optimization. Willey

Cordeau, J-F., Gendreau, M., Laporte, G., Potvin, J.-Y., & Semet, F. 2002. A guide to vehicle routing heuristics. Journal of the Operational Research Society, 53,512-522

Gendreau, M.,Hertz, A.,Laporte, G. 1994 “A Tabu Search Heuristic for the Vehicle Routing Problem”. Management Science, 40, 1276-1290.

Gillet, B.E, Miller, L.R.1974 “A heuristic algorithm for the vehicle dispatch problem” Operations Research 22, 240-349.

Glover,F.1989 “Tabu Search – parte 1”. ORSA Journal on Computing v.1, n.3.

Glover,F.,Laguna,M..1997 Tabu Search. Kluwer Academic Publishers.

Ho, S.C.,Haugland, D. 2004 “A tabu search heuristic for the vehicle routing problem with time windows and split deliveries” Computers & Operations Research 31, 1947-1964

Laporte, G. 1992. “The Vehicle Routing Problem: An overview of exact and approximate algorithms”. European Journal of operational Research 59,345-458

Laporte,G., Gendreau,M., Potvin,J., Semet, F.2000 “Classical and modern heuristics for the vehicle routing problem” Intl.Trans. in Op. Res 7,285-300

Lentra,J.K, Rinnoy K., G. 1981 “Complexity of Vehicle Routing and Scheduling Problems” Networks 11,221-227

Mole,R.H, Jamenson, S.R. 1976 “A sequential route - building algorithm employing a generalized savings criterion” Operations Research Quarterly 27,503-511

Osman, I; Laporte,G.1996 Metaheuristics: A bibliography. Annals of Operations Research 63, 513-628.

Pureza V.M., França, P.M. 1991.“Vehicle routinh problems via tabu search metaheuristic, Publication CRT-747, Centre de recherché sur les transports, Montreal.

Rego, C. 1998 “A Subpath Ejection Method for the Vehicle Routing Problem”, Management Science, 44, 1447-1459.

Tarantilis, C.D; Ioannou, G; Prastacos, G. 2005 “Advanced vehicle routing algorithms for complex operations management problems” Journal of Food Engineerig, 70:455-471.

Thangiah, S.R., Petrovik, P. 1997 Introduction to Genetic Heuristics and vehicle Routing Problems with Complex Constraints.In: Woodruff, David, L. Advances in Computacional and Stochastic Optimization, Logic programming , and Heuristic search: Interfaces in Computer Science and Operations research. Kluwer Academic Publishers.

Toth, P., Vigo, D. 2003 “Models, relaxations and exact approachs for the capacitated vehicle routing problem” Discrete Applied Mathematics 123, 487-512

Tyagi, M. 1968 “A Pratical Method for the Truck Dispatching Problem”. J. of the Operations Research Society of Japan, 10,76-92

Xu, J., Kelly, James P. 1996 “A Network Flow-Based Tabu Search Heuristic for the Vehicle Routing Problem” Transportation Science 30, 379-393

Willard, A.G. “Vehicle routinh using r-optimal tabu search”.MSc.Disseration, The Management School, Imperial College, London (1989)

GLOBAL ASYMPTOTIC VELOCITY OBSERVATION OF NONLINEAR SYSTEMS

Application to a Frictional Industrial Emulator

R. Guerra[‡], C. Iurian[⋆], L. Acho[‡⋆]
[‡]*Centro de Investigación y Desarollo de Technología Digital (CITEDI-IPN), Mexico*
[⋆]*Universitat Politècnica de Catalunya, Matemàtica Aplicada III, EUETIB, Barcelona, Spain*
inge_guerra@hotmail.com, {*claudiu.iurian, leonardo.acho*}*@upc.edu*

F. Ikhouane[†] and J. Rodellar[§]
[†]*Universitat Politècnica de Catalunya, Matemàtica Aplicada III, EUETIB, Barcelona, Spain*
[§]*Universitat Politècnica de Catalunya, Matemàtica Aplicada III, Barcelona, Spain*
{*faycal.ikhouane, jose.rodellar*}*@upc.edu*

Keywords: Velocity observers, friction, mechanical systems.

Abstract: Development of velocity observers for mechanical systems with friction deserves a special emphasis, because as evidenced in numerical and experimental tests when a state-of-the-art observer is armed, friction can induce high-frequency oscillations in the estimated signal. In this short paper, two new velocity-observation algorithms are designed, based on this previously reported observer, that eliminate the high-frequency oscillations noted. Numerical and experimental performance comparisons are carried through making use of LuGre model and a mechanical PID control system that incorporates the estimated velocity into the feedback loop.

1 INTRODUCTION

Velocity-dependent control laws such as PD, PID, and most robust control laws, among many others, theoretically require direct access to velocity. In reality, there are many applications in which this is not available either due to considerable manufacturing savings in cost, weight, and volume, or because the velocity measurements are highly contaminated with noise. In the latter case, for instance when measuring robot joint velocities, it may not even be desirable to do so (Arteaga and Kelly, 2004). Consequently, if the full-state information is missing, it is necessary to estimate the unmeasurable velocity through the use of an observer and feed it back into the controller. Such is the case with the frictional industrial emulator ECP model 220 used in our experiments. It has been documented when studying mechanical closed-loop control systems, that friction causes tracking errors, limit-cycles, and stick-slip motions, among other difficulties and usually unwanted phenomena (Armstrong-Hétlouvry et al., 1994). As evidenced in (Canudas de Wit and Fixot, 1991), (Canudas de Wit and Fixot, 1992), (Berghuis and Nijmeijer, 1993), and (Arteaga and Kelly, 2004) velocity observer design is a topic that has been and continues to be extensively studied. State observation of nonlinear inexact plants has been treated by utilizing discontinuous observers (Choi et al., 1999), (Xiong and Saif, 2001), and (Xian et al., 2004). However, little research focused on velocity observation of mechanical systems with friction at low velocities which, when incorporating existing observers, exhibit high-frequency oscillations in the velocity-estimation signal. A state-of-the-art, globally asymptotic, discontinuous velocity-estimation scheme for second-order mechanical systems has been presented in (Xian et al., 2004). Though very reliable, this observer is not specialized for mechanical systems in the presence of friction. Thus, a high-frequency component of the velocity observation is detected when numerical simulations and experimental testing are performed. The two newly proposed observers claimed here are an attempt to alleviate this unwanted oscillatory effect and try offer an increase in the observation reliability.

The remainder of this document proceeds as follows: Section 2 postulates two modified observers based on the velocity estimator stated in (Xian et al., 2004). The next section presents numerical simulations evidencing the performance of the observers. Section 4 begins with a description of the frictional experimental testbed, the ECP industrial emulator

(ECP, 1995), and illustrates the results obtained when implementing of the original observer as well as the two modifications proposed in the PID mechatronic system. Finally, conclusions are drawn in Section 5.

2 VELOCITY OBSERVATION

Consider a class of mechanical systems expressed by (Xian et al., 2004)

$$\ddot{x} = h(x,\dot{x}) + G(x,\dot{x})u, \tag{1}$$

where $x \in \mathbb{R}$ is the system output, $u(t) \in \mathbb{R}$ is the control input, and $h(x,\dot{x}) \in \mathbb{R}$ as well as $G(x,\dot{x}) \in \mathbb{R}$ are nonlinear functions[1]. The system (1) satisfies the following assumptions (Xian et al., 2004).

Assumption A1. Both $h(x,\dot{x})$ and $G(x,\dot{x})$ are C^1 functions.

Assumption A2. The control input is a C^1 function and $u(t), \dot{u}(t) \in L_\infty$.

Assumption A3. The system state is bounded for all time; *i.e.*, $x(t), \dot{x}(t) \in L_\infty$.

The goal of the velocity observer is to estimate the unmeasurable velocity signal $\dot{x}(t)$ using only-position measurement and assuming that $h(x,\dot{x})$, $G(x,\dot{x})$ and $u(t)$ are unknown (Xian et al., 2004). Let $\hat{\dot{x}}(t) \in \mathbb{R}$ be the estimated velocity and $\tilde{\dot{x}} = \dot{x} - \hat{\dot{x}}$ the velocity estimation error. Then, the objective of the velocity observer is to ensure that $\hat{\dot{x}}(t)$ converges to zero as the time tends to infinity.

Consider the following velocity observer (Xian et al., 2004)

$$\begin{aligned} \dot{\hat{x}} &= p + k_0\tilde{x}, \\ \dot{p} &= k_1 sgn(\tilde{x}) + k_2\tilde{x}, \end{aligned} \tag{2}$$

where k_0, k_1, and k_2 are positive constants, and $sgn(\cdot)$ is the signum function.

Remark 1. *System (2) can be expressed as*

$$\begin{aligned} \dot{y} &= f(y,x), \quad y \in R^2, \quad x \in R \\ z &= g(y,x), \quad z \in R \end{aligned}$$

where $y = [\hat{x} \quad p]^T$, $f(y,x) = [p + k_0\tilde{x} \quad k_1 sgn(\tilde{x}) + k_2\tilde{x}]^T$, $g(y,x) = p + k_0\tilde{x}$, *and* $z = \hat{\dot{x}}$ *is the output velocity estimation.*

To state the main result in (Xian et al., 2004), let

$$N_0(x,\dot{x},t) = h(x,\dot{x}) + G(x,\dot{x})u(t).$$

[1]Without loss of generality, we have assumed a one-degree-of-freedom (DOF) mechanical system.

Theorem 1 (Xian et al., 2004). *The velocity observer (2) ensures global asymptotic regulation of* $\dot{\tilde{x}}(t)$ *(i.e.,* $\dot{\tilde{x}}(t) \to 0$ *as* $t \to \infty$*) provided that* k_1 *satisfies*

$$k_1 > ||N_0(x,\dot{x},t)||_\infty + ||\dot{N}_0(x,\dot{x},t)||_\infty.$$

For detailed proof, see Theorem 2 in (Xian et al., 2004). Let us now put forward the following observer

$$\begin{aligned} \dot{\hat{x}} &= p + k_0\tilde{x}, \\ \dot{p} &= -k_1 sgn(\tilde{x}) + k_2\tilde{x}. \end{aligned} \tag{3}$$

Theorem 2. *The velocity observer (3) ensures global asymptotic regulation of* $\dot{\tilde{x}}(t)$ *(that is,* $\dot{\tilde{x}}(t) \to 0$ *as* $t \to \infty$*) provided that* k_1 *satisfies exactly the same conditions as in Theorem 1.*

Proof. Identical to that of Theorem 1.

Another alternative proposal to the original observer is the following

$$\begin{aligned} \dot{\hat{x}} &= p + k_0\tilde{x}, \\ \dot{p} &= k_1 sgn(\hat{x}) + k_2\tilde{x}. \end{aligned} \tag{4}$$

Theorem 3. *The velocity observer (4) also ensures global asymptotic regulation of* $\dot{\tilde{x}}(t)$ *(i.e.,* $\dot{\tilde{x}}(t) \to 0$ *as* $t \to \infty$*) on the condition that* k_1 *satisfies the same restriction as in Theorem 1.*

Proof. The same case as for the proof of Theorem 2 above.

These innovative observers are very similar to the one in Theorem 1, but they present significant differences. The observer in Theorem 2 introduces an inversion of the sign in the estimation dynamic that produces a filtering effect. The observer in Theorem 3 brings forward an estimation error $\hat{x}$ inside the signum function, with the intent of reducing the high-frequency content the error signal $\tilde{x}$ present in (2).

3 NUMERICAL EXPERIMENTS

Consider a linear motion of unit mass

$$\ddot{x} = u - f, \tag{5}$$

where f is the friction force. Assuming for a moment that $f = 0$ and $k_1 = 10$, the requirement imposed by all theorems is satisfied. We complete the velocity observers by setting $k_0 = k_2 = 10$. We also construct the following PID controller, which makes the closed-loop system globally asymptotically stable (if $f = 0$)

$$u = -k_p(x - x_d) - k_i \int (x - x_d)dt - k_d\dot{x}, \tag{6}$$

with $k_v = 6$, $k_p = 3$, $k_i = 4$, and the position reference set at $x_d = 1$ [m], see details in (Canudas de Wit et al., 1995). Because friction force can produce limit cycles within the system when the control law has an integrating action (Canudas de Wit et al., 1995), we incorporate a friction force in (5) by invoking the LuGre model and its standard parameters given in (Canudas de Wit et al., 1995); *i.e.,* f is obtained as a nonlinear dynamic. Simulation results are depicted in Figure 1 where the position and velocity of the system are pictured. These plots are a recreation of the experiment presented in (Canudas de Wit et al., 1995). Obviously, the PID controller (6) incorporates velocity measurement, which for this simulation was assumed to be available.

At this point, we repeated the previous simulation using the true velocity in the control law and implementing the three observers solely to estimate the velocity, obtaining the results shown in Figure 2. Since in the experimental application the velocity would not be available for use, the simulation was again repeated, this time invoking the observers from Theorems 1, 2, and 3 in the control law (*i.e.*, replacing $\dot{x}$ by its corresponding $\hat{\dot{x}}$ in (6)). The results are portrayed in Figures 3, 4, and 5, respectively. Note that chattering appears with the observer from Theorem 1 and not with the others. It is important to stress that chattering is undesirable in a physical system because the high-frequency switching can damage the system, as well as activate unmodelled dynamics (G. Bartolini, 1998) and (Hung, 1993).

Let us return to the previously considered case where the PID controller (6) is employed using the exact velocity, which is only observed. If we slightly modify the observer parameter values to $k_1 = 5$ and $k_0 = k_2 = 1$, the velocity observation obtained with the observer in (2) becomes highly oscillatory whereas the two observers proposed still provide good results, as shown in Figure 6. If furthermore we now feed the corresponding velocity estimation into the PID controller (6), the first observer (stated in Theorem 1) is much more sensible to variations in its parameters. This causes instability of the closed-control-loop (simulation pictures were then omitted for the first case); nevertheless, the other two new observers yield acceptable good simulation results, see Figures 7 and 8.

4 APPLICATION TO AN INDUSTRIAL EMULATOR

To have a more realistic comparison among the observers stated in Section 2, we proceed to implement them on an experimentation testbed.

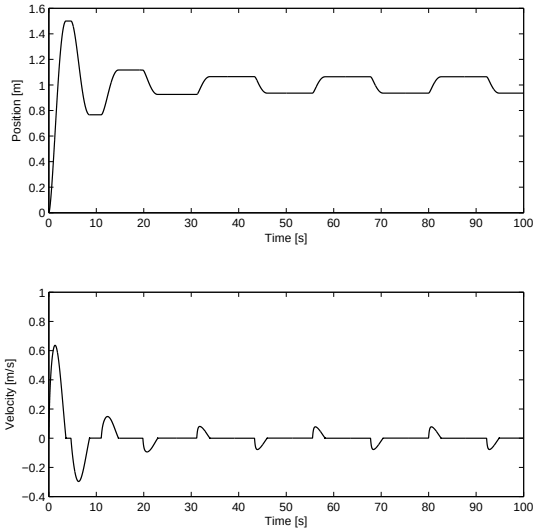


Figure 1: PID position control of a second-order system that incorporates Lugre friction model - see (Canudas de Wit et al., 1995).

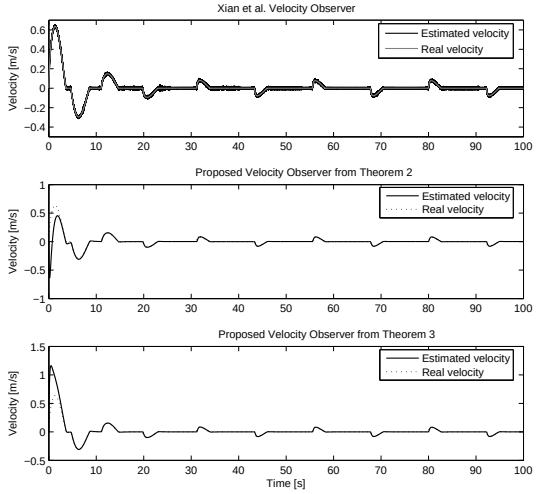


Figure 2: Comparison of the three estimators (for $k_{0..} = k_1 = k_2 = 10$): 1) Top with Theorem 1, 2) Middle with Theorem 2, and 3) Bottom with Theorem 3.

4.1 Experimental Setup

The experiments were performed on an ECP Model 220 industrial emulator which includes a PC-based control platform and a DC brushless servo system (ECP, 1995). The mechatronic system includes two motors, one as servo actuator and the other as disturbance input (not used here), a power amplifier, and two encoders which provide accurate position

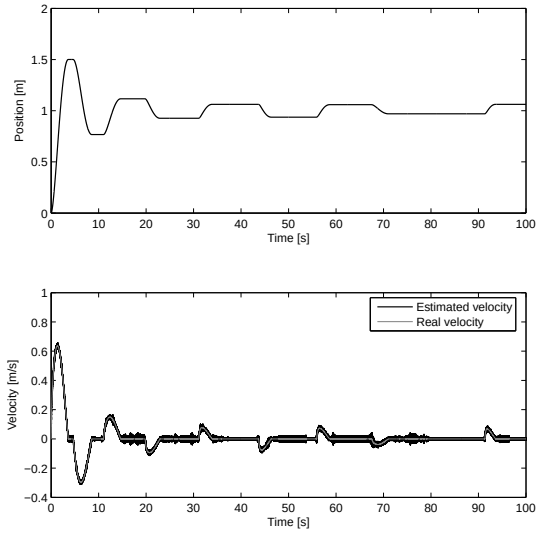


Figure 3: PID control incorporating velocity estimation from Theorem 1.

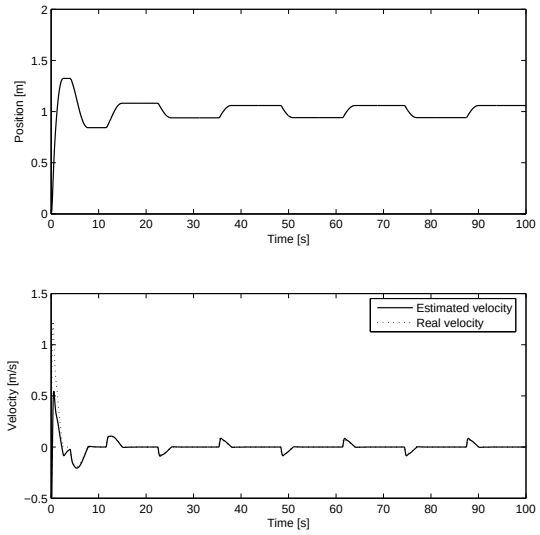


Figure 4: PID control employing velocity estimation given in Theorem 2.

measurements; i.e., 4000 lines per revolution with $4\times$ hardware interpolation giving 16000 counts per revolution to each encoder; 1 count (equivalent to 0.000392 radians or 0.0225 degrees) is the lowest angular position measurable (ECP, 1995). The system was set up to incorporate inertia and friction brake. The drive and load disks were connected via a $4:1$ speed reduction (see Figure 9). In order to demonstrate that the system is subject to the notorious effects of friction, we calculated according to the procedure described by (R. Kelly and Campa,

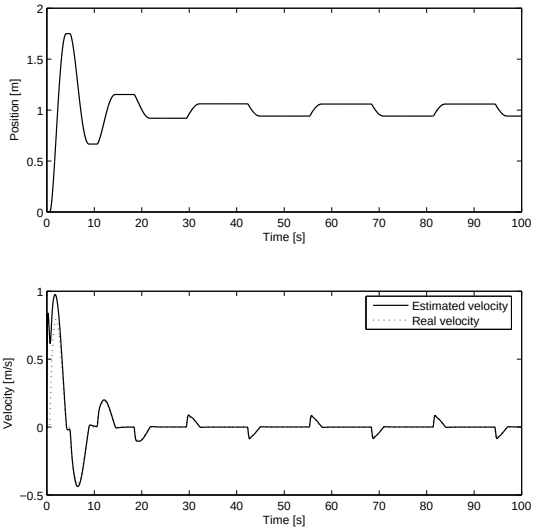


Figure 5: PID control utilizing the observer stated in Theorem 3.

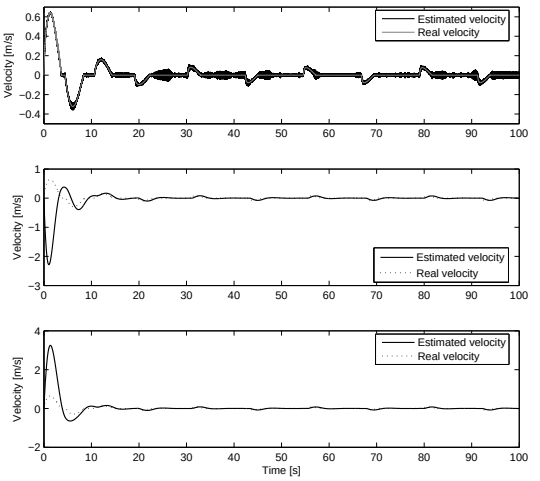


Figure 6: Comparison of the three estimators (for $k_{0..} = k_2 = 1$ and $k_1 = 5$): 1) Top with Theorem 1, 2) Middle with Theorem 2, and 3) Bottom with Theorem 3.

2000) the following friction coefficients for the system: $F_v = 0.05772[Nmsec/rad]$ (viscous friction coefficient) and $F_c = 0.43032[Nm]$ (Coulomb friction).

A Pentium 4, 2.80 GHz CPU, 512 MB RAM, computer running under Windows XP is programmed to implement the controller together with the interface medium ECP USR Executive 5.1, a C-like programming language (ECP, 1995). The system contains a data-acquisition board for digital to analog conversion and a counter board to read the position encoder outputs into the servo system. The minimum servo-loop closure sampling time T_s is 0.884 ms.

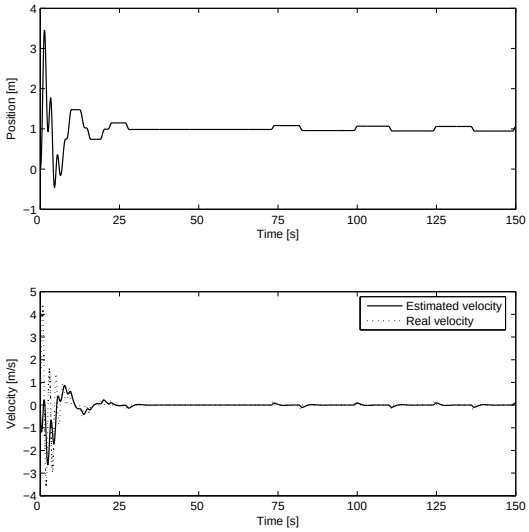


Figure 7: PID control containing the velocity as given by Theorem 2.

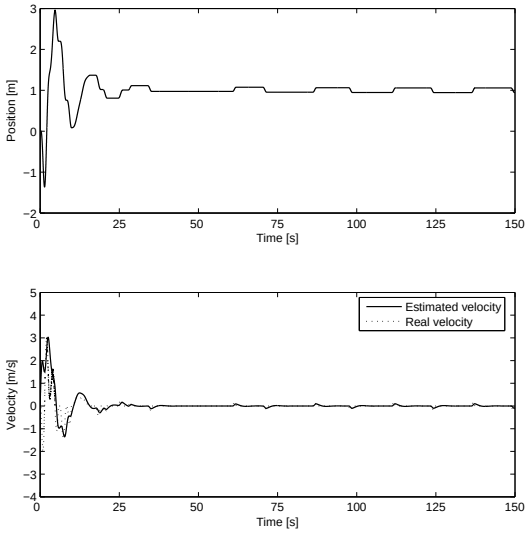


Figure 8: PID control making use of the observer in Theorem 3.

The output voltage signal generated by the system is in the range of $\pm 5V$ and is delivered to the motor drive via the DAC, the measurement feedback is a position signal (in counts or radians), measured at the shaft of each of the two disks by the optical rotary incremental position encoders, then it is read by the microcomputer by means of the counter board and delivered into the PC. A software interface has been built to easily transfer the raw data collected from the plant (by means of the ECP USR Executive program) to the Matlab workspace environment, in order to

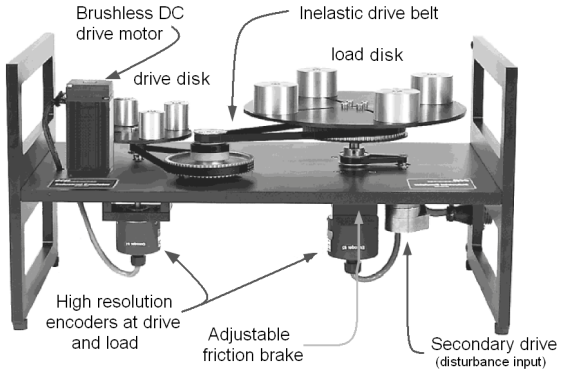


Figure 9: Mechanical system with friction.

display the results. The load disk is weighted with 4 masses of 0.50 kg each (at a radius of r = 10.0 cm) while the drive disk remains unweighted (see Figure 9). It is worth mentioning that the mechanical system has encoders which give accurate position measurements, nevertheless no direct velocity sensing is available (ECP, 1995). In this scenario, we implemented the aforementioned velocity observers, obtaining the results pictured in Figures 10 – 12.

4.2 Experimental Results

The implemented control law is as follows

$$u = -k_p(x - x_d) - k_i \int (x - x_d)dt - k_d\dot{\hat{x}},$$

with $k_d = 0.0011$, $k_p = 0.135$, and $k_i = 0.4$. The desired reference position was set to $x_d = 100\,[counts] = 0.0392\,[rad] = 2.25\,[deg]$.

As it can be noted in Figure 10, when the observer stated in Theorem 1 is employed, this produces an oscillation of relatively high-frequency and amplitude into the system. The observer (4), after the transient, eliminates both the amplitude and the frequency of this oscillation (Figure 11). The estimator from Theorem 3 further reduces the amplitude and duration of the transient, even more than observer (4), eliminating chattering as seen in Figure 12. Moreover, note that the position limit cycles in Figures 11–12, caused by friction in PID control of servo drives (Canudas de Wit et al., 1995), have the same rectangular-like waveform pattern as in Figure 1, and can be distinctly identified.

5 CONCLUSIONS

Two new velocity-observation designs are presented, and experimentally validated, for use in mechanical

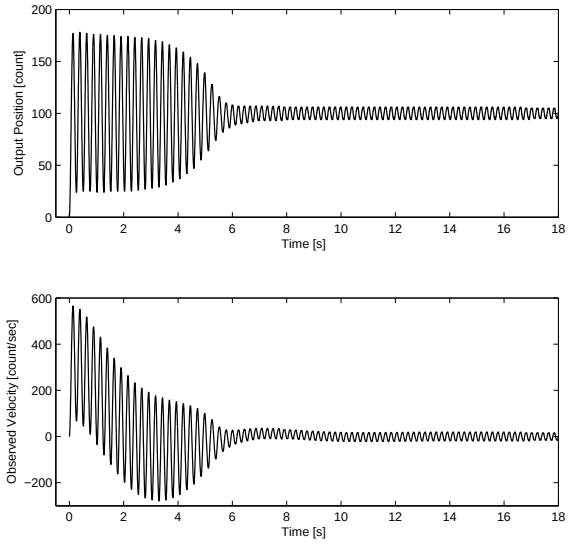


Figure 10: Experimental results using the observer from Theorem 1.

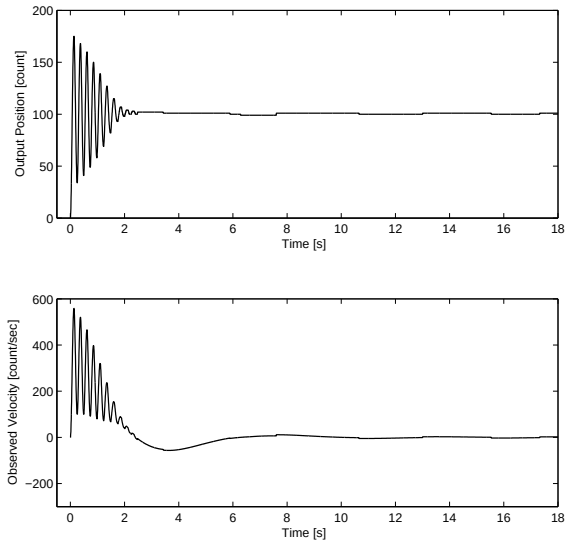


Figure 11: Experimental results employing the observer from Theorem 2.

systems with friction where only-position measurements are available. As it can be appreciated from numerical and experimental results, the proposed observer schemes are more efficient than their precursor in that chattering is eliminated from the velocity-observed signal. It is worth emphasizing that the presented observers (3) and (4) are especially interesting for industrial purposes, for they assure that the velocity-acquisition hardware can, without difficulty, be replaced by an analogous inexpensive software performing the same function.

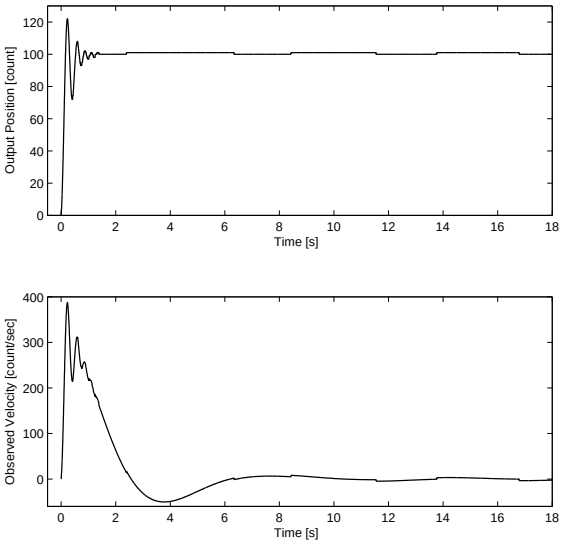


Figure 12: Experimental results utilizing the observer from Theorem 3.

ACKNOWLEDGEMENTS

This work was supported by CICYT through Grant DPI2005-08668-C03-01. The work of F. Ikhouane was supported by Spanish Ministry of Science and Education under “Ramón y Cajal” Program.

REFERENCES

Armstrong-Hétlouvry, B., Dupont, P., and Canudas de Wit, C. (1994). A survey of models, analysis tools and compensation methods for the control of machines with friction. *Automatica*, 30(7):1083–1138.

Arteaga, M. A. and Kelly, R. (2004). Robot control without velocity measurements: new theory and experimental results. *IEEE Trans. Robot. Automat*, 20(2):297–308.

Berghuis, H. and Nijmeijer, H. (1993). Global regulation of robots using only position measurements. In *Systems and Control Letters*. vol. 21, no. 4, pp. 289–293.

Canudas de Wit, C. and Fixot, N. (1991). Robot control via robust estimated state feedback. In *IEEE Trans. on Aut. Control*. vol. 36, no. 12, pp. 1497–1501.

Canudas de Wit, C. and Fixot, N. (1992). Adaptive control of robot manipulators via velocity estimated feedback. In *IEEE Trans. on Aut. Control*. vol. 37, no. 8, pp. 1234–1237.

Canudas de Wit, C., Olsson, H., Åström, K. J., and Lischinsky, P. (1995). A new model for control of systems with friction. In *IEEE Trans. on Aut. Control*. vol. 40, no. 3, pp. 419–425.

Choi, J., Misawa, E., and Young, G. (1999). A study on sliding mode state estimation. *J. dyn. syst. meas. control*, 121:255–260.

ECP (1995). Manual for model 220 industrial emulator/servo trainer. In *Educational Control Products*. California 91367, USA.

G. Bartolini, A. Ferrara, E. U. (1998). Chattering avoidance by second-order sliding mode control. In *IEEE Trans. on Automatic Control*. vol. 43, no. 2, pp. 241–246.

Hung, J. C. (1993). Chattering handling for variable structure control systems. In *Proc. of the IECON '93*. vol. 3, pp. 1968–1972, Maui, Hawaii, USA.

R. Kelly, J. L. and Campa, R. (2000). A measurement procedure for viscous and coulomb friction. In *IEEE Trans. Instrum. Meas.* vol. 49, no. 4, pp. 857–861.

Xian, C., de Queiroz, M. S., Dawson, D. M., and McIntyre, M. L. (2004). A discontinuous output controller and velocity observer for nonlinear mechanical systems. In *Automatica*. 40, pp. 695–700.

Xiong, Y. and Saif, M. (2001). Sliding mode observer for nonlinear uncertain systems. *IEEE Trans. on Aut. Control*, 46(12):2012–2017.

NONLINEAR MODEL PREDICTIVE CONTROL OF A LINEAR AXIS BASED ON PNEUMATIC MUSCLES

Harald Aschemann and Dominik Schindele
Chair of Mechatronics, University of Rostock, Justus-von-Liebig-Weg 6, D-18059 Rostock, Germany
Harald.Aschemann@uni-rostock.de, Dominik.Schindele@uni-rostock.de

Keywords: Mechatronics, predictive control, flatness-based methods, pneumatic muscle.

Abstract: This paper presents a nonlinear optimal control scheme for a mechatronic system that consists of a guided carriage and an antagonistic pair of pneumatic muscles as actuators. Modelling leads to a system of nonlinear differential equations including polynomial approximations of the volume characteristic as well as the force characteristic of the pneumatic muscles. The proposed control has a cascade structure: the nonlinear norm-optimal control of both pneumatic muscle pressures is based on an approximative solution of the corresponding HJB-equation, whereas the outer control loop involves a multivariable NMPC of the carriage position and the mean internal pressure of the pneumatic muscles. To improve the tracking behaviour, the feedback control loops are extended with nonlinear feedforward control based on differential flatness. Remaining model uncertainties as well as nonlinear friction can be counteracted by an observer-based disturbance compensation. Experimental results from an implementation on a test rig show an excellent control performance.

1 INTRODUCTION

Pneumatic muscle actuators are tension actuators consisting of a fiber-reinforced vulcanised rubber tubing with connection flanges at both ends. Due to a special fiber arrangement, the pneumatical muscle contracts with increasing internal pressure. This effect can be used for actuation purposes. Pneumatic muscles offer major advantages in comparison to classical pneumatic cylinders: significantly less weight, no stick-slip effects, insensitivity to dirty working environment, and a larger maximum force. The nonlinear characteristics of the muscle, however, demand for nonlinear control, e.g. flatness-based control (Fliess et al., 1995), (Aschemann and Hofer, 2004). For the practical investigation of control approaches the test rig shown in figure 1 has been built. Two guideways with roller bearing units allow for rectilinear movements of a carriage with small nonlinear friction forces. On opposite sides of the carriage, pneumatic muscles are arranged in an antagonistic configuration between the carriage and the rigid frame. The mass flow of air of each pneumatic muscle is controlled by means of a proportional valve. The in-flowing air is provided at a maximum pressure of 7 bar, whereas the out-flowing air is discharged at atmospheric pressure, i.e. 1 bar. Pressure declines in the case of large commanded mass flows are counteracted by using compensator reservoirs, which maintain an approximately constant pressure supply level for each pneumatic muscle. Similarly, an additional compensator reservoir in combination with a sound absorber reduces the noise caused by discharged air. The paper is structured as follows: first, the modelling of the mechatronic system is addressed. Second, a nonlinear cascade control scheme is proposed: nonlinear H_2-optimal control loops, which can be realised with high bandwidth, guarantee a specified internal pressure in each pneumatic muscle (Aschemann et al., 2006). The outer control loop involves nonlinear model predictive trajectory control of the carriage position and the mean muscle pressure as controlled variables and provides the reference pressures for the inner pressure control loops. Aiming at good tracking behaviour, feedforward control based on differential flatness is considered in the control structure as well. A disturbance force resulting from remaining modelling errors w.r.t. the force characteristic of the pneumatic

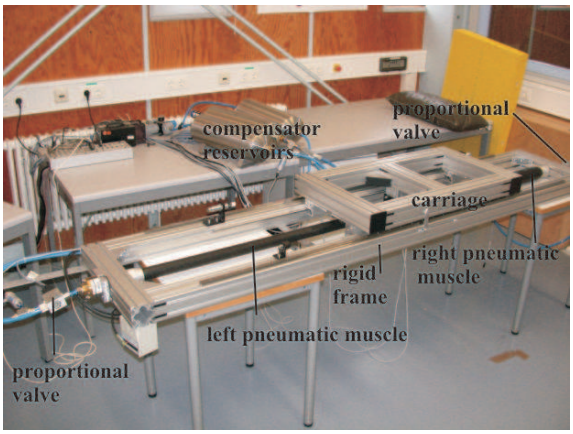


Figure 1: Linear axis test rig.

muscles as well as the friction characteristic of the carriage is estimated by a reduced-order disturbance observer and used for compensation in the nonlinear control scheme. By this, desired trajectories for both carriage position and mean pressure can be tracked with high accuracy as shown by experimental results from an implementation at the test rig.

2 SYSTEM MODELLING

As for modelling, the mechatronic system is divided in a pneumatic subsystem and a mechanical subsystem, which are coupled by the tension forces of the two pneumatic muscles. In contrast to the model of (Carbonell et al., 2001) the dynamics of the pneumatic subsystem is also included. The tension force $F_{Mi} \geq 0$ and the volume V_{Mi} of the pneumatic muscle i, $i = r, l$, nonlinearly depend on the according internal pressure p_{Mi} as well as the contraction length $\Delta\ell_{Mi}$. The origin of the generalised coordinate $x_S(t)$ of the carriage is defined as the position where the right muscle is fully contracted. Then, the constraints

$$\Delta\ell_{Ml}(x_S) = x_S, \;\; \Delta\ell_{Mr}(x_S) = \Delta\ell_{M,\max} - x_S \tag{1}$$

hold. Consequently, the contraction lengths of both pneumatic muscles are related to the carriage position.

2.1 Modelling of the Pneumatic Subsystem

The dynamics of the internal muscle pressure follows directly from a mass flow balance in combination with the energy equation for the compressed air in the muscle. As the internal muscle pressure is limited by a maximum value of $p_{Mi,max} = 7\ bar$, the ideal gas equation represents an accurate description of the thermodynamic behaviour. The thermodynamic process is modelled as a polytropic change of state with $n = 1.26$ as identified polytropic exponent. The identified volume characteristic of the pneumatic muscle can be described by a polynomial function

$$V_{Mi}\left(\Delta\ell_{Mi}, p_{Mi}\right) = \sum_{j=0}^{3} a_j \cdot \Delta\ell_{Mi}^{j} \cdot \sum_{k=0}^{1} b_k \cdot p_{Mi}^{k}. \tag{2}$$

with the contraction length $\Delta\ell_{Mi}$ and the muscle pressure p_{Mi}. The resulting state equation for the internal muscle pressure in the muscle i is given by

$$\dot{p}_{Mi} = \frac{n}{V_{Mi} + n \cdot \frac{\partial V_{Mi}}{\partial p_{Mi}} \cdot p_{Mi}} \left[R_L \cdot \vartheta(\cdot) \cdot \dot{m}_{Mi} - \frac{\partial V_{Mi}}{\partial \Delta\ell_{Mi}} \cdot \frac{\partial \Delta\ell_{Mi}}{\partial x_S} \cdot p_{Mi} \cdot \dot{x}_S\right], \tag{3}$$

where R_L denotes the gas constant of air. The function $\vartheta(n, T_S, T_{Mi}, sign(\dot{m}_{Mi}))$, which depends on the polytropic exponent n, the air supply temperature T_S, the internal temperature T_{Mi}, and the sign of the mass flow rate $\dot{m}_{Mi}$, can be approximated with good accuracy by the constant temperature T_0 of the ambiance. Thereby, temperature measurements can be avoided, and the implementational effort is significantly reduced.

2.2 Modelling of the Mechanical Subsystem

The mechanical subsystem is related to the motion of the carriage with mass $m_S = 30\ kg$ on its guideways. The nonlinear force characteristic $F_{Mi}(p_{Mi}, \Delta\ell_{Mi})$ of a pneumatic muscle represents the resulting tension force for given internal pressure p_{Mi} as well as given contraction length $\Delta\ell_{Mi}$. It has been identified by static measurements and, then, approximated by a polynomial description

$$F_{Mi}(\Delta\ell_{Mi}, p_{Mi}) = \bar{F}_{Mi}(\Delta\ell_{Mi}) \cdot p_{Mi} - f_{Mi}(\Delta\ell_{Mi}). \tag{4}$$

The equation of motion follows directly from Newton's second law as a second order differential equation for the carriage position

$$m_S \cdot \ddot{x}_S = F_{Ml}\left(\cdot\right) - F_{Mr}\left(\cdot\right) - F_U. \tag{5}$$

At this, remaining model uncertainties are taken into account by the disturbance force F_U. These uncertainties stem from approximation errors concerning the static muscle force characteristics, non-modelled viscoelastic effects of the vulcanised rubber material, and viscous damping as well as friction forces acting on the carriage.

3 NORM-OPTIMAL CONTROL OF THE MUSCLE PRESSURES

In the sequel, the nonlinear norm-optimal control design is presented (Lukes, 1969). The design approach applies to the following class of systems

$$\begin{aligned}\dot{\boldsymbol{x}}(t) &= \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t) + \boldsymbol{f}\left(\boldsymbol{x}(t), \boldsymbol{u}(t)\right), \\ \boldsymbol{y} &= \boldsymbol{h}\left(\boldsymbol{x}(t), \boldsymbol{u}(t)\right) = \boldsymbol{C}\boldsymbol{x}(t) + \boldsymbol{D}\boldsymbol{u}(t),\end{aligned} \tag{6}$$

with the affine control input $\boldsymbol{u} \in \mathcal{U} \subset \mathbb{R}^m$, the state vector $\boldsymbol{x} \in \mathcal{X} \subset \mathbb{R}^n$ and the output vector $\boldsymbol{y} \in \mathcal{Y} \subset \mathbb{R}^p$. The non-linearity $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})$ can be stated as $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) = \boldsymbol{f}^{(2)}(\boldsymbol{x}, \boldsymbol{u}) + \boldsymbol{f}^{(3)}(\boldsymbol{x}, \boldsymbol{u}) + ...$, where $\boldsymbol{f}^{(m)}(\boldsymbol{x}, \boldsymbol{u})$ denotes a polynomial of degree m in terms of $\boldsymbol{x}$ and $\boldsymbol{u}$. The H_2-optimal control aims at calculating a nonlinear state feedback law $\boldsymbol{u}(\boldsymbol{x})$ with $\boldsymbol{u}(\boldsymbol{0}) = \boldsymbol{0}$ such that the nonlinear cost function

$$J(\boldsymbol{u}) = \inf_{\boldsymbol{u} \in L_2^m[0,\infty)} \int_0^\infty \Lambda_2(\boldsymbol{x}, \boldsymbol{u}) \mathrm{d}t \tag{7}$$

with

$$\begin{aligned}\Lambda_2(\boldsymbol{x}, \boldsymbol{u}) =& \frac{1}{2}\left(\boldsymbol{y}^T\boldsymbol{Q}\boldsymbol{y} + \boldsymbol{u}^T\boldsymbol{R}\boldsymbol{u}\right) + l(\boldsymbol{x}, \boldsymbol{u}) \\ =& \frac{1}{2}\boldsymbol{x}^T\tilde{\boldsymbol{Q}}\boldsymbol{x} + \frac{1}{2}\boldsymbol{u}^T\tilde{\boldsymbol{R}}\boldsymbol{u} + \boldsymbol{x}^T\boldsymbol{N}\boldsymbol{u} + l(\boldsymbol{x}, \boldsymbol{u})\end{aligned} \tag{8}$$

is minimized. The symmetric, positive definite weighting matrix $\boldsymbol{Q} = \boldsymbol{Q}^T > 0$ accounts for the output variables, whereas the symmetric, positive definite weighting matrix $\boldsymbol{R}$ regards the control inputs. The nonlinear term $l(\boldsymbol{x}, \boldsymbol{u}) = l^{(3)}(\boldsymbol{x}, \boldsymbol{u}) + l^{(4)}(\boldsymbol{x}, \boldsymbol{u}) + ...$ consists of expressions of third or higher degrees in terms of $\boldsymbol{x}$ and $\boldsymbol{u}$. In the case of $l(\boldsymbol{x}, \boldsymbol{u}) = 0$ the cost function becomes quadratic. The solution of the above stated optimization problem is given by the positive definite solution $\mathcal{J}(\boldsymbol{x}) : \mathcal{X} \rightarrow \mathbb{R}^+$ of a nonlinear partial differential equation, the Hamilton-Jacobi-Bellman-equation (HJB-equation)

$$\min_{\boldsymbol{u}} H\left(\boldsymbol{x}, \mathcal{J}_x(\boldsymbol{x}), \boldsymbol{u}\right) = 0, \tag{9}$$

with the corresponding Hamiltonian

$$H = \mathcal{J}_x(\boldsymbol{x})\left[\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} + \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})\right] + \Lambda_2(\boldsymbol{x}, \boldsymbol{u}). \tag{10}$$

In the considered unconstrained case, the optimal solution $\boldsymbol{u}^*$ is obtained from the stationary condition $H_u = 0$. Consequently, the two equations

$$\begin{aligned}H =& \mathcal{J}_x(\boldsymbol{x})\left[\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} + \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})\right] + \frac{1}{2}\boldsymbol{x}^T\tilde{\boldsymbol{Q}}\boldsymbol{x} \\ &+ \boldsymbol{x}^T\boldsymbol{N}\boldsymbol{u} + \frac{1}{2}\boldsymbol{u}^T\tilde{\boldsymbol{R}}\boldsymbol{u} + l(\boldsymbol{x}, \boldsymbol{u}) = 0\end{aligned} \tag{11}$$

and

$$\begin{aligned}H_u =& \mathcal{J}_x(\boldsymbol{x})\left(\boldsymbol{B} + \boldsymbol{f}_u(\boldsymbol{x}, \boldsymbol{u})\right) + \boldsymbol{x}^T\boldsymbol{N} \\ &+ \boldsymbol{u}^T\tilde{\boldsymbol{R}} + l_u(\boldsymbol{x}, \boldsymbol{u}) = 0\end{aligned} \tag{12}$$

need to be solved. These equations are approximately solved by an approach according to (Lukes, 1969) based on power series expansions of the involved nonlinear functions . The gradient of the optimal solution $\mathcal{J}(\boldsymbol{x})$ and the control law are recursively determined by a step-by-step solution of both equations for a considered degree k of the according polynomials in $\boldsymbol{x}$ and $\boldsymbol{u}$.

3.1 Feedback Control Design

The control design for both internal muscle pressures is identical (Aschemann et al., 2006). For the sake of simplicity, the internal muscle pressure as state variable is denoted as $x := p_{Mi}$, $i = \{r, l\}$, and the control input as $u := u_{pi} = R_L \cdot T_0 \cdot \dot{m}_{Mi}$. Then, the state equation (3) can be rewritten as

$$\dot{x} = \frac{n \cdot \left(u - k_1(x_S, \dot{x}_S) \cdot x - k_2(x_S, \dot{x}_S) \cdot x^2\right)}{k_3(x_S) + k_4(x_S) \cdot x}, \tag{13}$$

According to the continuous dependence of the coefficients $k_i(x_S, \dot{x}_S)$ on x_S and $\dot{x}_S$, the resulting feedback control law is adapted by gain-scheduling. After a truncated Taylor series expansion with respect to $x = p_{Mi}$, the nonlinear state space description for the muscles pressure becomes

$$\begin{aligned}\dot{x} &= a(u) + b(u) \cdot x + c(u) \cdot x^2 + d(u) \cdot x^3 \\ &\quad + e(u) \cdot x^4 + g(u) \cdot x^5, \\ y &= x,\end{aligned} \tag{14}$$

The quadratic cost function with $l(x, u) = 0$ is given by

$$J(u) = \frac{1}{2}\int_0^\infty \left(q_p x^2 + r_p u^2\right) \mathrm{d}t, \tag{15}$$

where the scalar r_p serves as a weighting factor for the input variable $u = u_{pi}$ and the scalar q_p as weighting factor for the state variable, i.e. the muscle pressure $x = p_{Mi}$. The H_2-optimal control laws for these fast inner control loops are calculated up to the degree $k = 3$, i.e. $u_{pi,FB}(x) = \sum_{j=1}^{3} u^{0(j)}$. The resulting nonlinear feedback control law $u_{pl,FB}(p_{Ml}, x_S, \dot{x}_S)$ for the left muscle is depicted in Fig. 2 for $\dot{x}_S = 0.1$ m/s. Obviously, the linear part dominates the nonlinear terms.

3.2 Feedforward Control Design

As for feedforward control design, differential flatness can be exploited for the system under consideration (Fliess et al., 1995). The muscle pressure

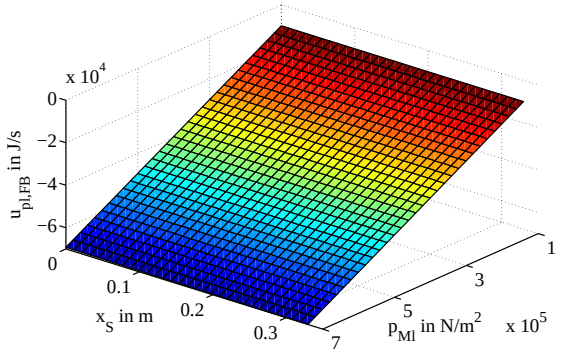


Figure 2: Nonlinear H_2-optimal feedback control law $u_{pl,FB}(p_{Ml}, x_S, \dot{x}_S)$ for the left pneumatic muscle for the gain-scheduling parameters x_S and $\dot{x}_S = 0.1$ m/s.

$y = p_{Mi}$, $i = \{r, l\}$, obviously represents a flat output of the corresponding inner control loop. Evaluating the state equation (3) for the muscle pressure with desired values for the flat output $y_d = p_{Mid}$ as well as its time derivative $\dot{y}_p = \dot{p}_{Mid}$ and solving for the feedforward control part $u_{pi,FF}$ result in

$$u_{pi,FF} = \left(\frac{V_{Mi}}{n} + \frac{\partial V_{Mi}}{\partial p_{Mi}} \cdot p_{Mid} \right) \dot{p}_{Mid} + \frac{\partial V_{Mi}}{\partial \Delta l_{Mi}} \frac{\partial \Delta l_{Mi}}{\partial x_S} \cdot \dot{x}_S \cdot p_{Mid} - u_{pi,FB}(p_{Mid}). \tag{16}$$

Note that the measured values x_S and $\dot{x}_S$ are used for a gain-scheduled adaptation of the feedforward control law. As a result, the overall control law for the inner control loops becomes $u_{pi} = u_{pi,FF} + u_{pi,FB}$.

4 NONLINEAR MPC

The main idea of the control approach consists in a minimization of a future tracking error in terms of the predicted state vector based on the actual state and the desired state vector resulting from trajectory planning (Lizarralde et al., 1999), (Jung and Wen, 2004). The minimization is achieved by repeated approximate numerical optimization in each time step, in the given case using the Newton-Raphson technique. The optimization is initialised in each time step with the optimization result of the preceeding time step in form of the input vector. The NMPC-algorithm is based on the following nonlinear discrete-time state space representation

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k) \ , \ \boldsymbol{y}_k = \boldsymbol{h}(\boldsymbol{x}_k, \boldsymbol{u}_k) \ , \tag{17}$$

with the state vector $\boldsymbol{x}_k \in \mathbb{R}^n$, the control input $\boldsymbol{u}_k \in \mathbb{R}^m$ and the output vector $\boldsymbol{y}_k \in \mathbb{R}^p$. The constant M specifies the prediction horizon T_P as a multiple of the sampling time t_s, i.e. $T_P = M \cdot t_s$. The predicted input vector at time k becomes

$$\boldsymbol{u}_{k,M} = \left[{\boldsymbol{u}_1^{(k)}}^T, ..., {\boldsymbol{u}_M^{(k)}}^T \right]^T \ , \tag{18}$$

with $\boldsymbol{u}_{k,M} \in \mathbb{R}^{m \cdot M}$. The predicted state vector at the end of the prediction horizon $\boldsymbol{\phi}_M(\boldsymbol{x}_k, \boldsymbol{u}_{k,M})$ is obtained by repeated substitution of k by $k+1$ in the discrete-time state equation (17)

$$\begin{aligned} \boldsymbol{x}_{k+2} &= \boldsymbol{f}(\boldsymbol{x}_{k+1}, \boldsymbol{u}_{k+1}) = \boldsymbol{f}(\boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k), \boldsymbol{u}_{k+1}) \\ &\vdots \\ \boldsymbol{x}_{k+M} &= \underbrace{\boldsymbol{f}(\cdots \boldsymbol{f}}_{M}(\boldsymbol{x}_k, \underbrace{\boldsymbol{u}_k), \cdots, \boldsymbol{u}_{k+M-1}}_{M}) \\ &= \boldsymbol{\phi}_M(\boldsymbol{x}_k, \boldsymbol{u}_{k,M}) \ . \end{aligned} \tag{19}$$

The difference of $\boldsymbol{\phi}_M(\boldsymbol{x}_k, \boldsymbol{u}_{k,M})$ and the desired state vector $\boldsymbol{x}_d$ leads to the final control error

$$\boldsymbol{e}_{M,k} = \boldsymbol{\phi}_M(\boldsymbol{x}_k, \boldsymbol{u}_{k,M}) - \boldsymbol{x}_d \ , \tag{20}$$

i.e. to the control error at the end of the prediction horizon. The cost function to be minimized follows as

$$J_{MPC} = \frac{1}{2} \cdot \boldsymbol{e}_{M,k}^T \boldsymbol{e}_{M,k} \ , \tag{21}$$

and, hence, the necessary condition for an extremum can be stated as

$$\frac{\partial J_{MPC}}{\partial \boldsymbol{e}_{M,k}} = \boldsymbol{e}_{M,k} \stackrel{!}{=} 0 \ . \tag{22}$$

A Taylor-series expansion of (22) at $\boldsymbol{u}_{k,M}$ in the neighbourhood of the optimal solution leads to the following system of equations

$$\boldsymbol{0} = \boldsymbol{e}_{M,k} + \frac{\partial \boldsymbol{\phi}_M}{\partial \boldsymbol{u}_{k,M}} \Delta \boldsymbol{u}_{k,M} + T.h.O. \tag{23}$$

The vector $\Delta \boldsymbol{u}_{k,M}$ denotes the difference which has to be added to the input vector $\boldsymbol{u}_{k,M}$ to obtain the optimal solution. The n equations (23) represent an under-determined set of equations with $m \cdot M$ unknowns having an infinite number of solutions. A unique solution for $\Delta \boldsymbol{u}_{k,M}$ can be determined by solving the following L_2-optimization problem with (23) as side condition

$$J = \frac{1}{2} \cdot \Delta \boldsymbol{u}_{k,M}^T \Delta \boldsymbol{u}_{k,M} + \boldsymbol{\lambda}^T \left(\boldsymbol{e}_{M,k} + \frac{\partial \boldsymbol{\phi}_M}{\partial \boldsymbol{u}_{k,M}} \Delta \boldsymbol{u}_{k,M} \right) \ . \tag{24}$$

Consequently, the necessary conditions can be stated as

$$\begin{aligned} \frac{\partial J}{\partial \Delta \boldsymbol{u}_{k,M}} \stackrel{!}{=} 0 &= \Delta \boldsymbol{u}_{k,M} + \left(\frac{\partial \boldsymbol{\phi}_M}{\partial \boldsymbol{u}_{k,M}} \right)^T \boldsymbol{\lambda}, \\ \frac{\partial J}{\partial \boldsymbol{\lambda}} \stackrel{!}{=} 0 &= \boldsymbol{e}_{M,k} + \frac{\partial \boldsymbol{\phi}_M}{\partial \boldsymbol{u}_{k,M}} \Delta \boldsymbol{u}_{k,M}, \end{aligned} \tag{25}$$

which leads to $\boldsymbol{e}_{M,k}$:

$$\boldsymbol{e}_{M,k} = \underbrace{\frac{\partial \boldsymbol{\phi}_M}{\partial \boldsymbol{u}_{k,M}} \left(\frac{\partial \boldsymbol{\phi}_M}{\boldsymbol{u}_{k,M}} \right)^T}_{\boldsymbol{S}(\boldsymbol{\phi}_M, \boldsymbol{u}_{k,M})} \boldsymbol{\lambda} \,. \tag{26}$$

If the matrix $\boldsymbol{S}\left(\boldsymbol{\phi}_M, \boldsymbol{u}_{k,M}\right)$ is invertible, the vector $\boldsymbol{\lambda}$ can be calculated

$$\boldsymbol{\lambda} = \boldsymbol{S}^{-1}\left(\boldsymbol{\phi}_M, \boldsymbol{u}_{k,M}\right) \boldsymbol{e}_{M,k} \,. \tag{27}$$

An almost singular matrix $\boldsymbol{S}\left(\boldsymbol{\phi}_M, \boldsymbol{u}_{k,M}\right)$ can be treated by a modification of (27)

$$\boldsymbol{\lambda} = \left[\mu \boldsymbol{I} + \boldsymbol{S}\left(\boldsymbol{\phi}_M, \boldsymbol{u}_{k,M}\right)\right]^{-1} \boldsymbol{e}_{M,k} \,, \tag{28}$$

where $\boldsymbol{I}$ denotes the unity matrix. The regularisation parameter $\mu > 0$ in (28) may be chosen constant or may be calculated by a sophisticated algorithm. The latter solution improves the convergence of the optimization but increases, however, the computational complexity. Solving (25) for $\Delta \boldsymbol{u}_{k,M}$ and inserting $\boldsymbol{\lambda}$ according to (27) or (28), directly leads to the L_2-optimal solution

$$\begin{aligned} \Delta \boldsymbol{u}_{k,M} &= -\left(\frac{\partial \phi_M}{\partial \boldsymbol{u}_{k,M}} \right)^T \boldsymbol{S}^{-1}\left(\boldsymbol{\phi}_M, \boldsymbol{u}_{k,M}\right) \boldsymbol{e}_{M,k} \\ &= -\left(\frac{\partial \phi_M}{\partial \boldsymbol{u}_{k,M}} \right)^{\dagger} \boldsymbol{e}_{M,k} \,. \end{aligned} \tag{29}$$

Here, $\left(\frac{\partial \phi_M}{\partial \boldsymbol{u}_{k,M}}\right)^{\dagger}$ denotes the Moore-Penrose pseudo inverse of $\frac{\partial \phi_M}{\partial \boldsymbol{u}_{k,M}}$. The overall NMPC-algorithm can be described as follows:
Choice of the initial input vector $\boldsymbol{u}_{0,M}$ at time $k = 0$, e.g. $\boldsymbol{u}_{0,M} = \boldsymbol{0}$, and repetition of steps a) - c) at each sampling time $k \geq 0$:

a) Calculation of an improved input vector $\boldsymbol{v}_{k,M}$ according to

$$\boldsymbol{v}_{k,M} = \boldsymbol{u}_{k,M} - \eta_k \left(\frac{\partial \boldsymbol{\phi}_M}{\partial \boldsymbol{u}_{k,M}} \right)^{\dagger} \boldsymbol{e}_{M,k} \,. \tag{30}$$

The step width η_k can be determined with, e.g., the Armijo-rule.

b) For the calculation of $\boldsymbol{u}_{k+1,M}$ the elements of the vector $\boldsymbol{v}_{k,M}$ have to be shifted by m elements and the steady-state input vector $\boldsymbol{u}_d$ corresponding to the final state has to be inserted at the end $\boldsymbol{u}_d$

$$\begin{aligned} \boldsymbol{u}_{k+1,M} = &\begin{bmatrix} \boldsymbol{0}_{(m(M-1)\times m)} \\ \boldsymbol{I}_{(m)} \end{bmatrix} \boldsymbol{u}_d \\ &+ \begin{bmatrix} \boldsymbol{0}_{(m(M-1)\times m)} & \boldsymbol{I}_{(m(M-1))} \\ \boldsymbol{0}_{m\times m} & \boldsymbol{0}_{(m\times m(M-1))} \end{bmatrix} \boldsymbol{v}_{k,M}. \end{aligned} \tag{31}$$

In general, the steady-state control input $\boldsymbol{u}_d$ can be computed from

$$\boldsymbol{x}_d = \boldsymbol{f}(\boldsymbol{x}_d, \boldsymbol{u}_d). \tag{32}$$

For differentially flat systems the desired input vector $\boldsymbol{u}_d$ is given by the inverse dynamics and can be stated as a function of the flat outputs and their time derivatives.

c) The first m elements of the improved input vector $\boldsymbol{v}_{k,M}$ are applied as control input at time k

$$\boldsymbol{u}_k = \begin{bmatrix} \boldsymbol{I}_{(m)} & \boldsymbol{0}_{(m\times m(M-1))} \end{bmatrix} \boldsymbol{v}_{k,M} \,. \tag{33}$$

In the proposed algorithm only one iteration is performed per time step. A similar approach using several iteration steps is described in (Weidemann et al., 2004).

4.1 Numerical Calculations

The analytical computation of the Jacobian $\frac{\partial \phi_M}{\partial \boldsymbol{u}_{k,M}}$ becomes increasingly complex for larger values of M. Therefore, a numerical approach is preferred taking advantage of the chain rule with $i = 0, ..., M-1$

$$\begin{aligned} \frac{\partial \phi_M}{\partial \mathbf{u}_{i+1}^{(k)}} = &\frac{\partial \phi_M}{\partial \mathbf{x}_{k+M-1}} \cdot \frac{\partial \mathbf{x}_{k+M-1}}{\partial \mathbf{x}_{k+M-2}} \cdot \ldots \\ &\cdot \frac{\partial \mathbf{x}_{k+i+2}}{\partial \mathbf{x}_{k+i+1}} \cdot \frac{\partial \mathbf{x}_{k+i+1}}{\partial \mathbf{u}_{i+1}^{(k)}} \,. \end{aligned} \tag{34}$$

Introducing the abbreviations

$$\boldsymbol{A}_i := \frac{\partial \boldsymbol{x}_{k+i+1}}{\partial \boldsymbol{x}_{k+i}} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}} \left(\boldsymbol{x}_{k+i}, \boldsymbol{u}_{i+1}^{(k)} \right) \,, \tag{35}$$

$$\boldsymbol{B}_i := \frac{\partial \boldsymbol{x}_{k+i+1}}{\partial \boldsymbol{u}_{i+1}^{(k)}} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}} \left(\boldsymbol{x}_{k+i}, \boldsymbol{u}_{i+1}^{(k)} \right) \,, \tag{36}$$

the Jacobian can be computed as follows

$$\begin{aligned} \frac{\partial \phi_M}{\partial \mathbf{u}_{k,M}} = &[\mathbf{A}_{M-1}\mathbf{A}_{M-2}\cdots\mathbf{A}_1\mathbf{B}_0, \\ &\mathbf{A}_{M-1}\cdots\mathbf{A}_2\mathbf{B}_1, ..., \mathbf{A}_{M-1}\mathbf{B}_{M-2}, \mathbf{B}_{M-1}] \,. \end{aligned} \tag{37}$$

For the inversion of the symmetric and positive definite matrix $\boldsymbol{S}\left(\boldsymbol{\phi}_M, \boldsymbol{u}_{k,M}\right) = \frac{\partial \boldsymbol{\phi}_M}{\partial \boldsymbol{u}_{k,M}} \left(\frac{\partial \boldsymbol{\phi}_M}{\partial \boldsymbol{u}_{k,M}} \right)^T$ the Cholesky-decomposition has proved advantageous in terms of computational effort.

4.2 Choice of the Nmpc Design Parameters

The most important NMPC design parameter is the prediction horizon T_P, which is given as the product of the sampling time t_s and the constant value

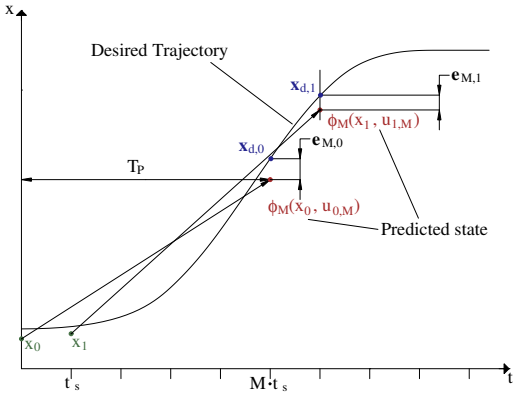


Figure 3: Design parameters.

M. Large values of T_P lead to a slow and smooth transient behaviour and result in a robust and stable control loop. For fast trajectory tracking, however, a smaller value T_P is desirable concerning a small tracking error. The choice of the sampling time t_s is crucial as well: a small sampling time is necessary regarding discretization error and stability; however, the NMPC-algorithm has to be evaluated in real-time within the sampling inverval. Furthermore, the smaller t_s, the larger becomes M for a given prediction horizon, which in turn increases the computational complexity of the optimization step. Consequently, a system-specific trade-off has to be made for the choice of M and t_s. This paper follows the moving horizon approach with a constant prediction horizon and, hence, a constant dimension $m \cdot M$ of the corresponding optimization problem in constrast to the shrinking horizon approach (Weidemann et al., 2004).

4.3 Input Constraints

One major advantage of predictive control is the possibility to easily account for input constraints, which are present in almost all control applications. To this end, the cost function can be extended with a corresponding term

$$h(\boldsymbol{u}_j^{(k)}) = \begin{cases} 0 & \boldsymbol{u}_{min} \leq \boldsymbol{u}_j^{(k)} \leq \boldsymbol{u}_{max} \\ g_1(\boldsymbol{u}_j^{(k)}) \text{ for } & \boldsymbol{u}_j^{(k)} > \boldsymbol{u}_{max} \\ g_2(\boldsymbol{u}_j^{(k)}) & \boldsymbol{u}_j^{(k)} < \boldsymbol{u}_{min} \end{cases} , \tag{38}$$

which has to be evaluated componentwise, i.e. for each input variable at each sampling time. Thus, the contribution of the additional input constraints depending on $\boldsymbol{u}_{k,M}$ is given by

$$z(\boldsymbol{u}_{k,M}) = \sum_{j=1}^{M} h(\boldsymbol{u}_j^{(k)}) \tag{39}$$

Instead of $\boldsymbol{e}_{M,k}$ the vector $\left[\boldsymbol{e}_{M,k}^T, z\right]^T$ has to be minimized in the NMPC-algorithm.

5 MODIFICATIONS OF THE ALGORITHM

To improve trajectory tracking behaviour, the NMPC-algorithm can be modified as follows:
(1) Instead of a minimization of the control error at the end of the prediction horizon given by the difference of the predicted value $\boldsymbol{\phi}_M(\boldsymbol{x}_k, \boldsymbol{u}_{k,M})$ and the according reference value $\boldsymbol{x}_d$, the minimization could take into account additional predicted errors $\boldsymbol{e}_{Mi,k} = \boldsymbol{\phi}_{Mi}(\boldsymbol{x}_k, \boldsymbol{u}_{k,Mi}) - \boldsymbol{x}_{di}$, $i \in \mathbb{N}$, $Mi < M$. Thus, the cost function (21) is modified as follows

$$J_{MPC} = \frac{1}{2}(\boldsymbol{e}_{M1,k}^T \boldsymbol{e}_{M1,k} + ... + \boldsymbol{e}_{M,k}^T \boldsymbol{e}_{M,k}) \tag{40}$$

The required values $\boldsymbol{\phi}_{Mi}$ are already known from the calculation of $\boldsymbol{\phi}_M$ and, hence, do not further increase the computational effort. Unfortunately, the additional computation of $\frac{\partial \boldsymbol{\phi}_{Mi}}{\partial \boldsymbol{u}_{Mi,k}}$ as well as the increased dimension of the matrix to be inverted $\boldsymbol{S}(\boldsymbol{\phi}_{M1}, ..., \boldsymbol{\phi}_M, \boldsymbol{u}_{k,M1}, ..., \boldsymbol{u}_{k,M})$ have a significant impact on the computation time. Therefore, the number of expressions in the cost function should be kept as small as possible, especially in the given case of a fast higher-dimensional system.
(2) A further improvement of the trajectory tracking behaviour can be achieved if an input vector resulting from an inverse system model is used as initial vector for the subsequent optimization step instead of the last input vector. Since the system under consideration is differentially flat (Aschemann and Hofer, 2004), the required ideal control input can be derived for a given reference trajectory. The slightly modified algorithm can be stated as follows

a) Calculation of the ideal input vector $\boldsymbol{u}_{k,M}^{(d)}$ by evaluating an inverse system model with the specified reference trajectory as well as a certain number $\beta \in \mathbb{N}$ of its time derivatives

$$\boldsymbol{u}_{k,M}^{(d)} = \boldsymbol{u}_{k,M}^{(d)} \left(\boldsymbol{y}_d, \dot{\boldsymbol{y}}_d, ..., \overset{(\beta)}{\boldsymbol{y}}_d \right) . \tag{41}$$

b) Calculation of the improved input vector $\boldsymbol{v}_{k,M}$ based on the equation

$$\boldsymbol{v}_{k,M} = \boldsymbol{u}_{k,M}^{(d)} - \eta_k \left(\frac{\partial \boldsymbol{\phi}_M}{\partial \boldsymbol{u}_{k,M}} \right)^{\dagger} \boldsymbol{e}_{M,k} . \tag{42}$$

c) Application of the first m elements of $\boldsymbol{v}_{k,M}$ to the process

$$\boldsymbol{u}_k = \left[\begin{array}{cc} \boldsymbol{I}_{(m)} & \boldsymbol{0}_{(m \times m(M-1))} \end{array} \right] \boldsymbol{v}_{k,M} \; . \quad (43)$$

If the reference trajectory is known in advance, the according reference input vector $\boldsymbol{u}_{k,M}^{(d)}$ can be computed offline. Consequently, the online computational time remains unaffected. Of course, all the proposed modifications could be combined.

5.1 Nmpc of the Carriage Position

The state space representation for the position control design can be directly derived from the equation of motion for the carriage

$$\dot{\boldsymbol{x}} = \left[\begin{array}{c} \dot{x}_S \\ \ddot{x}_S \end{array} \right] = \left[\begin{array}{c} \dot{x}_S \\ \frac{F_{Ml}(x_S, p_{Ml}) - F_{Mr}(x_S, p_{Mr})}{m_S} \end{array} \right] . \quad (44)$$

The carriage position x_S and the carriage velocity $\dot{x}_S$ represent the state variables, whereas the input vector consists of the left as well as the right internal muscle pressure, p_{Ml} and p_{Mr}. The discrete-time representation of the continous-time system (44) is obtained by Euler discretisation

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + t_s \cdot \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k) \quad (45)$$

Using this simple discretisation method, the computational effort for the NMPC-algorithm can be kept acceptable. Furthermore, no significant improvement was obtained for the given system with the Heun discretisation method because of the small sampling time $t_s = 5\ ms$. Only in the case of large sampling times, e.g. $t_s > 20\ ms$, the increased computational effort caused by a sophisticated time discretisation method is advantageous. Then, the smaller discretisation error allows for less time integration steps for a specified prediction horizon, i.e. a smaller number M. As a result, the smaller number of time steps can overcompensate the larger effort necessary for a single time step. The flat output variables of (44) are given by

$$\boldsymbol{y} = \left[\begin{array}{c} x_S \\ p_M \end{array} \right] = \left[\begin{array}{c} x_S \\ \frac{1}{2} \cdot (p_{Ml} + p_{Mr}) \end{array} \right] . \quad (46)$$

Using the desired trajectories for the carriage position x_{Sd} and the mean muscle pressure p_{Md}, the corresponding desired input values result in

$$\boldsymbol{u}_d = \left[\begin{array}{c} p_{Mld} \\ p_{Mrd} \end{array} \right] = \frac{1}{\bar{F}_{Ml}(\cdot) + \bar{F}_{Mr}(\cdot)} \cdot \left[\begin{array}{c} f_{Ml}(\cdot) - f_{Mr}(\cdot) + 2\bar{F}_{Mr}(\cdot) p_{Md} + m_S \ddot{x}_{Sd} \\ f_{Mr}(\cdot) - f_{Ml}(\cdot) + 2\bar{F}_{Ml}(\cdot) p_{Md} - m_S \ddot{x}_{Sd} \end{array} \right] . \quad (47)$$

5.2 Compensation of the Valve Characteristic and Disturbances

The nonlinear valve characteristic (VC) is compensated by pre-multiplying with its inverse valve characteristic (IVC) in each input channel. Here, the inverse valve characteristic depends both on the commanded mass flow and on the measured internal pressure. Disturbance behaviour and tracking accuracy in view of model uncertainties can be significantly improved by introducing a compensating control action provided by a reduced-order disturbance observer, which uses an integrator as disturbance model. The observer design is based on the equation of motion for the carriage (5), where the variable F_U takes into account both the friction force F_{RS} and the remaining model uncertainties of the muscle force characteristics ΔF_M, i.e. $F_U = F_{RS} - \Delta F_M$. Moreover, the disturbance observer is capable of counteracting impacts of changing carriage mass Δm_S as well, which results in $F_U = F_{RS} + \Delta m_S \cdot \ddot{x}_S - \Delta F_M$. As the complete state vector $\mathbf{x} = [x_S, \dot{x}_S]^T$ is forthcoming, the reduced-order disturbance observer yields a disturbance force estimate $\hat{F}_U$. Disturbance compensation is achieved by using the estimated force $\hat{F}_U$ as additional control action after an appropriate input transformation.

6 EXPERIMENTAL RESULTS

For the experiments at the linear axis test rig the synchronized reference trajectories for the carriage position as well as the mean muscle pressure depicted in the upper part of fig. 4 have been used. First, several changes are specified for the carriage position between $0.02\ m$ and $0.29\ m$ at a constant mean pressure of $4\ bar$. Second, the mean pressure is increased up to $5\ bar$ and kept constant during some subsequent fast position variations by $-0.02\ m$. Third, several larger position changes are performed with a constant mean pressure of $4\ bar$.

During trajectory tracking the number M is set to small values. The sampling time has been kept constant at $t_s = 5\ ms$. Fig. 4 shows the results obtained with the choice $M = 15, i.e.\ T_P = 75\ ms$. Smaller prediction horizons would lead to a tendency towards increasing oscillatory behaviour and, finally, to instability. During the acceleration and deceleration intervalls a maximum position control error $e_{x,max}$ of approx. $4\ mm$ occurs. The maximum control error of the mean pressure e_p is only slightly above an absolute value of approx. $0.12\ bar$. The importance of

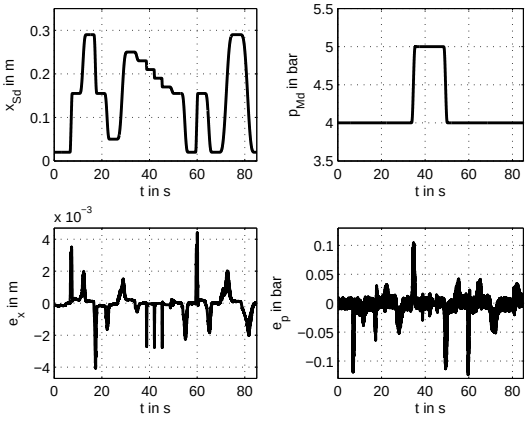


Figure 4: Reference trajectories and according tracking errors for carriage position and mean pressure ($T_P = 75\ ms$).

the observer-based disturbance compensation is emphasized by Fig. 5. Without this control part the maximum position control error increases up to approx. 7 mm.

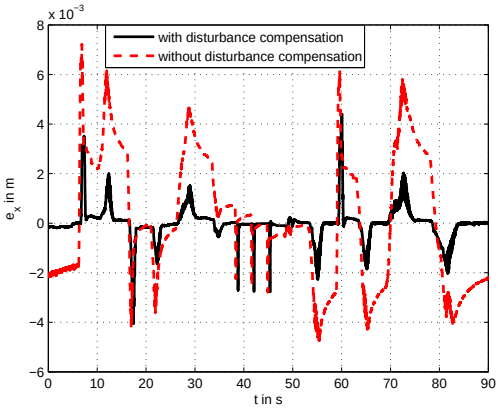


Figure 5: Position control error e_x with and without disturbance observer ($T_P = 75\ ms$).

7 CONCLUSIONS

In this paper, a cascaded trajectory control scheme using nonlinear optimal design is presented for a carriage driven by pneumatic muscles. The modelling of this mechatronic system leads to a system of four nonlinear differential equations. The nonlinear characteristics of the pneumatic muscles are approximated by polynomials. The nonlinearity of the valve is linearised by means of a pre-multiplication with its approximated inverse characteristic. The inner control loops of the cascade involve a norm-optimal control of the internal muscle pressure with high bandwidth. The outer nonlinear model predictive control loop is responsible for trajectory tracking with carriage position and mean pressure as controlled variables. Remaining model uncertainties are taken into account by a disturbance force estimated by means of a disturbance observer. Experimental results from an implementation on a test rig emphasise the excellent closed-loop performance with maximum position errors of $4\ mm$ during the movements, negligible steady-state position error and steady-state pressure error of less than $0.02\ bar$.

REFERENCES

Aschemann, H. and Hofer, E. (2004). Flatness-based trajectory control of a pneumatically driven carriage with model uncertainties. *Proceedings of NOLCOS 2004, Stuttgart, Germany*, pages 239 – 244.

Aschemann, H., Schindele, D., and Hofer, E. (2006). Nonlinear optimal control of a mechatronic system with pneumatic muscles actuators. *CD-ROM-Proceedings of MMAR 2006, Miedzyzdroje, Poland,*.

Carbonell, P., Jiang, Z. P., and Repperger, D. (2001). Comparative study of three nonlinear control strategies for a pneumatic muscle actuator. *Proceedings of NOLCOS 2001, Saint-Petersburg, Russia*, pages 167–172.

Fliess, M., Levine, J., Martin, P., and Rouchon, P. (1995). Flatness and defect of nonlinear systems: Introductory theory and examples. *Int. J. Control 61*, 6:1327 – 1361.

Jung, S. and Wen, J. (2004). Nonlinear model predictive control for the swing-up of a rotary inverted pendulum. *ASME J. of Dynamic Systems, Measurement and Control*, 126:666 – 673.

Lizarralde, F., Wen, J., and Hsu, L. (1999). A new model predictive control strategy for affine nonlinear control systems. *Proc of the American Control Conference (ACC '99), San Diego*, pages 4263 – 4267.

Lukes, D. (1969). Optimal regulation of nonlinear dynamical systems. *SIAM Journal Control*, 7:75 – 100.

Weidemann, D., Scherm, N., and Heimann, B. (2004). Discrete-time control by nonlinear online optimization on multiple shrinking horizons for underactuated manipulators. *Proceedings of the 15th CISM-IFToMM Symposium on Robot Design, Dynamics and Control, Montreal*.

RSRT: RAPIDLY EXPLORING SORTED RANDOM TREE

Online Adapting RRT to Reduce Computational Solving Time while Motion Planning in Wide Configuration Spaces

Nicolas Jouandeau
L.I.A.S.D.
Dept. MIME
Université Paris8
n@ai.univ-paris8.fr

Keywords: Motion-planning, soft computing.

Abstract: We present a new algorithm, named *RSRT*, for *Rapidly-exploring Random Trees* (*RRT*) based on inherent relations analysis between *RRT* components. *RRT* algorithms are designed to consider interactions between these inherent components. We explain properties of known variations and we present some future once which are required to deal with dynamic strategies. We present experimental results for a wide set of path planning problems involving a free flying object in a static environment. The results show that our *RSRT* algorithm (where *RSRT* stands for Rapidly-exploring Sorted Random Trees) is faster than existing ones. This results can also stand as a starting point of a motion planning benchmark instances which would make easier further comparative studies of path planning algorithms.

1 INTRODUCTION

Literally, planning is the definition of a sequence of orders which reach a previously selected goal. In a geometrical context, planning considers a workspace, an initial position, a final position and a set of constraints characterizing a mobile $\mathcal{M}$. The problem of planning could be resumed in two questions: the existence of a solution to a given problem and the definition of a solution to a problem that has at least one solution. In this paper, the problem of planning is focused on the second one, *i.e.* identifying solutions for problems that have at least one solution. The complexity of such a solution depends on the mobile workspace, its caracteristics (*i.e.* number of degrees of freedom) and the required answer complexity (*i.e.* the model and the local planner). Each dimension of these three parts contributes to define the problem dimension. Complexity is exponential in the problem dimension, so probabilistic methods propose to solve geometrical path-planning problems by finding a valid solution without guarantee of optimality. This particular relation to optimality associates probabilistic methods with problems known as difficult (also called non-deterministic polynomial in space (Canny, 1987)). In these methods, solving a path-planning problem consists in exploring the space in order to compute a solution with a determinist algorithm (Latombe, 1991). The specificity of these methods can be summarized with a random sampling of the search space, which reduces the determinist-polynomial complexity of the resolution (Schwartz and Sharir, 1983). The increase of computers capacities and the progress of the probabilistic methods, made solvable problems more complex during last decades. The principal alternatives of research space are the configuration space C (Lozano-Pérez, 1983), the state space X (Donald et al., 1993) and the state-time space ST (Fraichard, 1993). C is intended to motion planning in static environments. X adds differential constraints. ST adds the possibility of a dynamic environment. The concept of high-dimensional configuration spaces is initiated by J. Barraquand *et al.* (Barraquand and Latombe, 1990) to use a manipulator with 31 degrees of freedom. P. Cheng (Cheng, 2001) uses these methods with a 12 dimensional state space involving rotating rigid objects in $3D$ space. S. M. LaValle (LaValle, 2004) presents such a space with a hundred dimensions for either a robot manipulator or a couple of mobiles. The probabilistic methods mostly used in such spaces are *Randomized Path Planning* (*RPP*), *Probabilistic RoadMap* (*PRM*) and *Rapidly exploring Random*

Trees (*RRT*). The *RPP* method introduced by J. Barraquand *et al.* (Barraquand and Latombe, 1991) is a variation of the gradient method also introduced by O. Khatib (Khatib, 1985). Random moves make possible to escape from the local minima and recover the completeness. These random moves follow a Gaussian law. Each move is independent of the previous one. With obstacles, moves must remain in free space C_{free}. In case of collision, moves are reflected on the obstacles. The *PRM* method is introduced simultaneously by L.E. Kavraki *et al.* (Kavraki, 1995) and by P. Svestka *et al.* (Svestka, 1997) under the heading *Probabilistic Path Planner* (*PPP*). Their resolution principle divides the path planning problem into two successive stages: a *learning phase* that builds a graph and a *query phase* that builds a solution based on the previous graph. During the *learning phase*, the graph is built in C_{free} where each node is randomly selected according to a uniform distribution in C_{free}. This uniform distribution is justified by the need of exploring the entire free space. It is obtained by a random sampling associated to a collision detector. During the *query phase*, the graph is used to connect two configurations q_{init} and q_{obj} included in C_{free}. At each iteration, a local path planner seeks a way to connect a new node to the graph and also tries to connect q_{init} and q_{obj}. The *RRT* method introduced by S.M. LaValle (LaValle, 1998) is based on the construction of a tree T in the considered space $\mathcal{S}$. Starting from the initial position q_{init}, the construction of the tree is carried out by integrating control commands iteratively. Each iteration aims at bringing closer the mobile $\mathcal{M}$ to an element e randomly selected in $\mathcal{S}$. To avoid cycles, two elements e of T cannot be identical. In practice, *RRT* is used to solve various problems such as negotiating narrow passages made of obstacles (Ferré and Laumond, 2004), finding motions that satisfy obstacle-avoidance and dynamic balances constraints (Kuffner et al., 2003), making Mars exploration vehicles strategies (Williams et al.,), searching hidden objects (Tovar et al., 2003), rallying a set of points or playing hide-and-seek with another mobile (Simov et al., 2002) and many others mentioned in (LaValle, 2004). Thus by their efficiency to solve a large set of problems, the *RRT* method can be considered as the most general one.

In the next section, we present existing *RRT* algorithms.

2 RAPIDLY EXPLORING RANDOM TREES

In its initial formulation, *RRT* algorithms are defined without goal. The exploration tree covers the surrounding space and progress blindly towards free space.

A geometrical path planning problem aims generally at joining a final configuration q_{obj}. To solve the path planning problem, the *RRT* method searches a solution by building a tree (ALG. 1) rooted at the initial configuration q_{init}. Each node of the tree results from the mobile constraints integration. Its edges are commands that are applied to move the mobile from a configuration to another.

The *RRT* method is a random incremental search which could be casting in the same framework of Las Vegas Algorithms (*LVA*). It repeats successively a loop made of three phases: generating a random configuration q_{rand}, selecting the nearest configuration q_{prox}, generating a new configuration q_{new} obtained by numerical integration over a fixed time step Δt. The mobile $\mathcal{M}$ and its constraints are not explicitly specified. Therefore, modifications for additional constraints (such as non-holonomic) are considered minor in the algorithm formulation.

In this first version, C is presented without obstacle in an arbitrary space dimension. At each iteration, a local planner is used to connect each couples (q_{new}, q_{prox}) in C. The distance between two configurations in T is defined by the time-step Δt. The local planner is composed by temporal and geometrical integration constraints. The resulting solution accuracy is mainly due to the chosen local planner. k defines the maximum depth of the search. If no solution is found after k iterations, the search can be restarted with the previous T without re-executing the init function (ALG. 1 line 1).

The *RRT* method, inspired by traditional Artificial Intelligent techniques for finding sequences between an initial and a final element (*i.e.* q_{init} and q_{obj}) in a well-known environment, can become a bidirectional search (shortened *Bi-RRT* (LaValle and Kuffner, 1999)). Its principle is based on the simultaneous construction of two trees (called T_{init} and T_{obj}) in which the first grows from q_{init} and the second from q_{obj}. The two trees are developped towards each other while no connection is established between them. This bidirectional search is justified because the meeting configuration of the two trees is nearly the half-course of the configuration space separating q_{init} and q_{obj}. Therefore, the resolution time complexity is reduced (Russell and Norvig, 2003).

```
rrt(q_init, k, Δt, C)
1    init(q_init, T);
2    for i ← 1 à k
3        | q_rand ← randomState(C);
4        | q_prox ← nearbyState(q_rand, T);
5        | q_new ← newState(q_prox, q_rand, Δt);
6        | addState(q_new, T);
7        | addLink(q_prox, q_new, T);
8    return T;
```

ALG. 1: Basic *RRT* building algorithm.

RRT-Connect (Kuffner and LaValle, 2000) is a variation of *Bi-RRT* that consequently increase the *Bi-RRT* convergence towards a solution thanks to the enhancement of the two trees convergence. This has been settled to:

- ensure a fast resolution for "simple" problems (in a space without obstacle, the *RRT* growth should be faster than in a space with many obstacles);
- maintain the probabilistic convergence property. Using heuristics modify the probability convergence towards the goal and also should modify its evolving distribution. Modifying the random sampling can create local minima that could slow down the algorithm convergence.

```
connectT(q, Δt, T)
1    r ← ADVANCED;
2    while r = ADVANCED
3        | r ← expandT(q, Δt, T);
4    return r;
```

ALG. 2: Connecting a configuration q to a graph T with *RRT-Connect*.

In *RRT-Connect*, the two graphs previously called T_{init} and T_{obj} are called now T_a and T_b (ALG. 3). T_a (*respectively* T_b) replaces T_{init} and T_{obj} alternatively (*respectively* T_{obj} and T_{init}). The main contribution of *RRT-Connect* is the ConnectT function which move towards the same configuration as long as possible (*i.e.* without collision). As the incremental nature algorithm is reduced, this variation is designed for non-differential constraints. This is iteratively realized by the expansion function (ALG. 2). A connection is defined as a succession of successful extensions. An expansion towards a configuration q becomes either an extension or a connection. After connecting successfully q_{new} to T_a, the algorithm tries as many extensions as possible towards q_{new} to T_b. The configuration q_{new} becomes the convergence configuration q_{co} (ALG. 3 lines 8 and 10).

```
rrtConnect(q_init, q_obj, k, Δt, C)
1    init(q_init, T_a);
2    init(q_obj, T_b);
3    for i ← 1 à k
4        | q_rand ← randomState(C);
5        | r ← expandT(q_rand, Δt, T_a);
6        | if r ≠ TRAPPED
7        |       if r = REACHED
8        |           | q_co ← q_rand;
9        |       else
10       |           | q_co ← q_new;
11       |       if connectT(q_co, Δt, T_b) =
         |         REACHED
12       |           | sol ← plan(q_co, T_a, T_b);
13       |           | return sol;
14       | swapT(T_a, T_b);
15   return TRAPPED;
```

ALG. 3: Expanding two graphs T_a and T_b towards themselves with *RRT-Connect*. q_{new} mentionned line 10 correponds to the q_{new} variable mentionned line 9 ALG. 4.

Inherent relations inside the adequate construction of T in C_{free} shown in previous works are:

- the deviation of random sampling in the variations *Bi-RRT* and *RRT-Connect*. Variations include in *RRT-Connect* are called *RRT-ExtCon*, *RRT-ConCon* and *RRT-ExtExt*; they modify the construction strategy of one of the two trees of the method *RRT-Connect* by changing priorities of the extension and connection phases (LaValle and Kuffner, 2000).
- the well-adapted q_{prox} element selected according to its collision probability in the variation *CVP* and the integration of collision detection since q_{prox} generation (Cheng and LaValle, 2001).
- the adaptation of C to the vicinity accessibility of q_{prox} in the variation *RC-RRT* (Cheng and LaValle, 2002).
- the parallel execution of growing operations for n distinct graphs in the variation *OR parallel Bi-RRT* and the growing of a shared graph with a parallel q_{new} sampling in the variation *embarrassingly parallel Bi-RRT* (Carpin and Pagello, 2002).
- the sampling adaptation to the *RRT*

growth (Jouandeau and Chérif, 2004; Cortès and Siméon, 2004; Lindemann and LaValle, 2003; Lindemann and LaValle, 2004; Yershova et al., 2005).

By adding the collision detection in the given space S during the expansion phase, the selection of nearest neighbor q_{prox} is realized in $S \cap C_{free}$ (ALG. 4). Although the collision detection is expensive in computing time, the distance metric evaluation ρ is subordinate to the collision detector. U defines the set of admissible orders available to the mobile $\mathcal{M}$. For each expansion, the function expandT (ALG. 4) returns three possible values: REACHED if the configuration q_{new} is connected to T, ADVANCED if q is only an extension of q_{new} which is not connected to T, and TRAPPED if q cannot accept any successor configuration q_{new}.

```
expandT(q, Δt, T)
1     q_prox ← nearbyState(q, T);
2     d_min ← ρ(q_prox, q);
3     success ← FALSE;
4     foreach u ∈ U
5           | q_tmp ← integrate(q, u, Δt);
6           | if isCollisionFree(q_tmp, q_prox, M, C)
7                     d ← ρ(q_tmp, q_rand);
8                     if d < d_min
9                           | q_new ← q_tmp;
10                          | d_min ← d;
11                          | success ← TRUE;
12    if success = TRUE
13          | insertState(q_prox, q_new, T);
14          | if q_new = q
15                    return REACHED;
14          | return ADVANCED;
17    return TRAPPED;
```

ALG. 4: Expanding T with obstacles.

In the next section, we examine in detail justifications of our algorithm and the inherent relations in the various components used. This study enables us to synthesize a new algorithm named Rapidly exploring Sorted Random Tree (*RSRT*), based on reducing collision detector calls without modification of the classical random sampling strategy.

3 RSRT ALGORITHM

Variations of *RRT* method presented in the previous section is based on the following sequence :

- generating q_{rand};
- selecting q_{prox} in T;
- generating each successor of q_{prox} defined in U.
- realizing a colliding test for each successor previously defined;
- selecting a configuration called q_{new} that is the closest to q_{rand} among successors previously defined; This selected configuration has to be collision free.

The construction of T corresponds to the repetition of such a sequence. The collision detection discriminates the two possible results of each sequence:

- the insertion of q_{new} in T (*i.e.* without obstacle along the path between q_{prox} and q_{new});
- the rejection of each q_{prox} successors (*i.e.* due to the presence of at least one obstacle along each successors path rooted at q_{prox}).

The rejection of q_{new} induces an expansion probability related to its vicinity (and then also to q_{prox} vicinity); the more the configuration q_{prox} is close to obstacles, the more its expansion probability is weak. It reminds one of fundamentals *RRT* paradigm: free spaces are made of configurations that admit various number of available successors; good configurations admit many successors and bad configurations admit only few ones. Therefore, the more good configurations are inserted in T, the better the *RRT* expansion will be. The problem is that we do not previously know which good and bad configurations are needed during *RRT* construction, because the solution of the considered problem is not yet known. This problem is also underlined by the parallel variation *OR Bi-RRT* (Carpin and Pagello, 2002) (*i.e.* to define the depth of a search in a specific vicinity). For a path planning problem p with a solution s available after n integrations starting from q_{init}, the question is to maximize the probability of finding a solution; According to the concept of "rational action", the response of *P*3 class to adapt a on-line search can be solved by the definition of a formula that defines the cost of the search in terms of "local effects" and "propagations" (Russell, 2002). These problems find a way in the tuning of the behavior algorithm like *CVP* did (Cheng and LaValle, 2001).

In the case of a space made of a single narrow passage, the use of bad configurations (which successors

generally collide) is necessary to resolve such problem. The weak probability of such configurations extension is one of the weakness of the *RRT* method.

```
newExpandT(q, Δt, T)
1     q_prox ← nearbyState(q, T);
2     S ← ∅;
3     foreach u ∈ U
4           | q ← integrate(q_prox, u, Δt);
5           | d ← ρ(q, q_rand);
6           | S ← S + {(q, d)};
7     qsort(S, d);
8     n ← 0;
10    while n < Card(S)
11          | s ← getTupleIn(n, S);
12          | q_new ← firstElementOf(s);
13          | if isCollisionFree(q_new, q_prox, M, C)
14                    insertState(q_prox, q_new, T);
15                    if q_new = q
16                          | return REACHED;
17                    return ADVANCED;
18          | n ← n + 1;
19    return TRAPPED;
```

ALG. 5: Expanding T and reducing the collision detection.

To bypass this weakness, we propose to reduce research from the closest element (ALG. 4) to the first free element of C_{free}. This is realized by reversing the relation between collision detection and distance metric; the solution of each iteration is validated by subordinating collision tests to the distance metric; the first success call to the collision detector validates a solution. This inversion induces:

- a reduction of the number of calls to the collision detector proportionally to the nature and the dimension of U; Its goal is to connect the collision detector and the derivative function that produce each q_{prox} successor.
- an equiprobability expansion of each node independently of their relationship with obstacles;

The T construction is now based on the following sequence:

1. generating a random configuration q_{rand} in C;
2. selecting q_{prox} the nearest configuration to q_{rand} in T (ALG. 5 line 1);
3. generating each successors of q_{prox} (ALG. 5 lines 3 to 6); each successor is associated with its distance metric from q_{rand}. It produces a couple called s stored in S;
4. sorting s elements by distance (ALG. 5 lines 7);
5. selecting the first collision-free element of S and breaking the loop as soon as this first element is discovered (ALG. 5 lines 16 and 17);

4 EXPERIMENTS

This section presents experiments performed on a Redhat Linux Cluster that consists of 8 Dual Core processor 2.8 GHz Pentium 4 (5583 bogomips) with 512 MB DDR Ram.

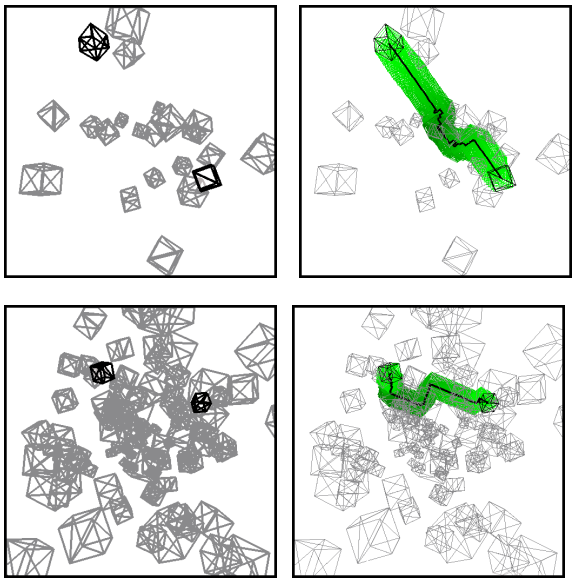


Figure 1: 20 obstacles problem and its solution (upper couple). 100 obstacles problem and its solution (lower couple).

To perform the run-time behavior analysis for our algorithm, we have generated series of problems that gradually contains more 3D-obstacles. For each problem, we have randomly generated ten different instances. The number of obstacles is defined by the sequence 20, 40, 60, ..., 200, 220. In each instance, all obstacles are cubes and their sizes are randomly varying between $(5,5,5)$ and $(20,20,20)$. The mobile is a cube with a fixed size $(10,10,10)$. Obstacles and mobile coordinates are varying between $(-100,-100,-100)$ and $(100,100,100)$. For each instance, a set of 120 q_{init} and 120 q_{obj} are generated in C_{free}. By combinating each q_{init} and each q_{obj}, 14400 configuration-tuples are available for each instance of each problem. For all that, our benchmark is made of more than 1.5 million problems. An instance with 20 obstacles is shown in FIG. 1 on the lower part and another instance with 100 obstacles in FIG. 1

on the left part. On these two examples, q_{init} and q_{obj} are also visible. We used the Proximity Query Package (*PQP*) library presented in (Gottschalk et al., 1996) to perform the collision detection. The mobile is a free-flying object controlled by a discretized command that contains 25 different inputs uniformly dispatched over translations and rotations. The performance was compared between *RRT-Connect* (using the *RRT-ExtCon* strategy) and our *RSRT* algorithm (ALG. 5).

The choice of the distance metric implies important consequences on configurations' connexity in C_{free}. It defines the next convergence node q_{co} for the local planner. The metric distance must be selected according to the behavior of the local planner to limit its failures. The local planner chosen is the straight line in C. To validate the toughness of our algorithm regarding to *RRT-Connect*, we had use three different distance metrics. Used distance metrics are:

- the Euclidean distance (mentioned *Eucl* in FIG. 2 to 4)

$$d(q,q') = \left(\sum_{k=0}^{i} (c_k - c'_k)^2 + nf^2 \sum_{k=0}^{j} (\alpha_k - \alpha'_k)^2 \right)^{\frac{1}{2}}$$

  where nf is the normalization factor that is equal to the maximum of c_k range values.

- the scaled Euclidean distance metric (mentioned *Eucl2* in FIG. 2 to 4)

$$d(q,q') = \left(s \sum_{k=0}^{i} (c_k - c'_k)^2 + nf^2(1-s) \sum_{k=0}^{j} (\alpha_k - \alpha'_k)^2 \right)^{\frac{1}{2}}$$

  where s is a fixed value 0.9;

- the Manhattan distance metric (mentioned *Manh* in FIG. 2 to 4)

$$d(q,q') = \sum_{k=0}^{i} \|c_k - c'_k\| + nf \sum_{k=0}^{j} \|\alpha_k - \alpha'_k\|$$

  where c_k are axis coordinates and α_k are angular coordinates.

For each instance, we compute the first thousand successful trials to establish average resolving times (FIG. 2), standard deviation resolving times (FIG. 3) and midpoint resolving times (FIG. 4). These trials are initiated with a fixed random set of seed. Those fixed seed assume that tested random suite are different between each other and are the same between instances of all problems. As each instance is associated to one thousand trials, each point of each graph is the average over ten instances (and then over ten thousands trials).

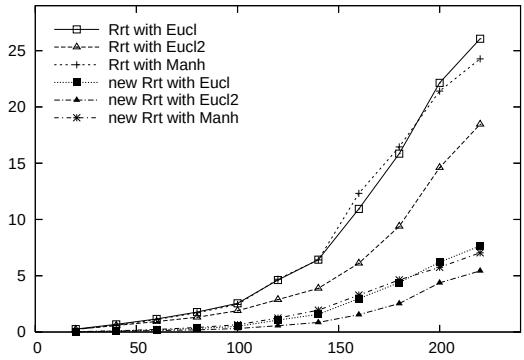


Figure 2: Averages resolving times.

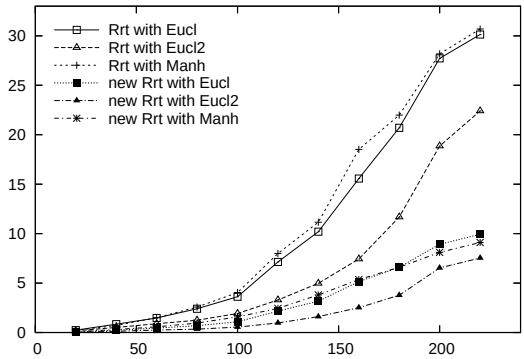


Figure 3: Standard deviation resolving times.

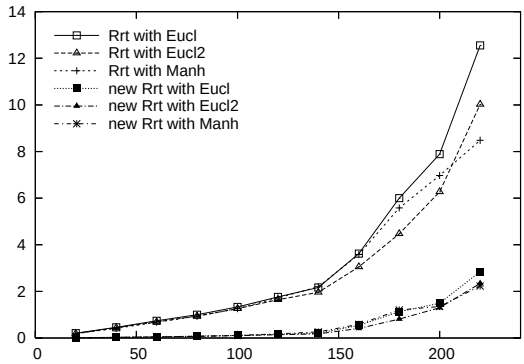


Figure 4: Midpoint resolving times.

On each graph, the number of obstacles is on x-axis and resolving time in *sec.* is on y-axis.

Figure 2 shows that average resolving time of our algorithm oscillates between 10 and 4 times faster

than the original *RRT-Connect* algorithm. As the space obstruction grows linearly, the resolving time of *RRT-Connect* grows exponentially while *RSRT* algorithm grows linearly. Figure 3 shows that the standard deviation follows the same profile. It shows that *RSRT* algorithm is more robust than *RRT-Connect*. Figure 4 shows that midpoints' distributions follow the average resolving time behavior. This is a reinforcement of the success of the *RSRT* algorithm. This assumes that half part of time distribution are 10 to 4 times faster than *RRT-Connect*.

5 CONCLUSION

We have described a new *RRT* algorithm, the *RSRT* algorithm, to solve motion planning problems in static environments. *RSRT* algorithm accelerates consequently the resulting resolving time. The experiments show the practical performances of the *RSRT* algorithm, and the results reflect its classical behavior. The results given above(have been evaluated on a cluster which provide a massive experiment analysis. The challenging goal is now to extend the benchmark that is proposed to every motion planning methods. The proposed benchmark will be enhanced to specific situations that allow *RRT* to deal with motion planning strategies based on statistical analysis.

REFERENCES

Barraquand, J. and Latombe, J. (1990). A Monte-Carlo Algorithm for Path Planning with many degrees of Freedom. In *Int. Conf. on Robotics and Automation (ICRA'90)*.

Barraquand, J. and Latombe, J. (1991). Robot motion planning: A distributed representation approach. *Int. Journal of Robotics Research (IJRR'91)*.

Canny, J. (1987). *The complexity of robot motion planning*. PhD thesis, Massachusetts Institute of Technology. Artificial Intelligence Laboratory.

Carpin, S. and Pagello, E. (2002). On Parallel RRTs for Multi-robot Systems. In *8th Conf. of the Italian Association for Artificial Intelligence (AI*IA'02)*.

Cheng, P. (2001). Reducing rrt metric sensitivity for motion planning with differential constraints. Master's thesis, Iowa State University.

Cheng, P. and LaValle, S. (2001). Reducing Metric Sensitivity in Randomized Trajectory Design. In *Int. Conf. on Intelligent Robots and Systems (IROS'01)*.

Cheng, P. and LaValle, S. (2002). Resolution Complete Rapidly-Exploring Random Trees. In *Int. Conf. on Robotics and Automation (ICRA'02)*.

Cortès, J. and Siméon, T. (2004). Sampling-based motion planning under kinematic loop-closure constraints. In *Workshop on the Algorithmic Foundations of Robotics (WAFR'04)*.

Donald, B., Xavier, P., Canny, J., and Reif, J. (1993). Kinodynamic Motion Planning. *Journal of the ACM*.

Ferré, E. and Laumond, J. (2004). An iterative diffusion algorithm for part disassembly. In *Int. Conf. Robotics and Automation (ICRA'04)*.

Fraichard, T. (1993). Dynamic trajectory planning with dynamic constraints: a "state-time space" approach. In *Int. Conf. Robotics and Automation (ICRA'93)*.

Gottschalk, S., Lin, M., and Manocha, D. (1996). Obb-tree: A hierarchical structure for rapid interference detection. In *Proc. of ACM Siggraph'96*.

Jouandeau, N. and Chérif, A. A. (2004). Fast Approximation to gaussian random sampling for randomized motion planning. In *Int. Symp. on Intelligent Autonomous Vehicules (IAV'04)*.

Kavraki, L. (1995). *Random networks in configuration space for fast path planning*. PhD thesis, Stanford University.

Khatib, O. (1985). Real-time obstacle avoidance for manipulators and mobile robots. In *Int. Conf. on Robotics and Automation (ICRA'85)*.

Kuffner, J. and LaValle, S. (2000). RRT-Connect: An efficient approach to single-query path planning. In *Int. Conf. on Robotics and Automation (ICRA'00)*.

Kuffner, J., Nishiwaki, K., Kagami, S., Inaba, M., and Inoue, H. (2003). Motion planning for humanoid robots. In *Int'l Symp. Robotics Research (ISRR'03)*.

Latombe, J. (1991). *Robot Motion Planning (4th edition)*. Kluwer Academic.

LaValle, S. (1998). Rapidly-exploring random trees: A new tool for path planning. Technical Report 98-11, Dept. of Computer Science, Iowa State University.

LaValle, S. (2004). *Planning Algorithms*. [on-line book]. http://msl.cs.uiuc.edu/planning/.

LaValle, S. and Kuffner, J. (1999). Randomized kinodynamic planning. In *Int. Conf. on Robotics and Automation (ICRA'99)*.

LaValle, S. and Kuffner, J. (2000). Rapidly-exploring random trees: Progress and prospects. In *Workshop on the Algorithmic Foundations of Robotics (WAFR'00)*.

Lindemann, S. and LaValle, S. (2004). Incrementally reducing dispersion by increasing Voronoi bias in RRTs. In *Int. Conf. on Robotics and Automation (ICRA'04)*.

Lindemann, S. R. and LaValle, S. M. (2003). Current issues in sampling-based motion planning. In *Int. Symp. on Robotics Research (ISRR'03)*.

Lozano-Pérez, T. (1983). Spatial Planning: A Configuration Space Approach. In *Trans. on Computers*.

Russell, S. (2002). Rationality and Intelligence. In Press, O. U., editor, *Common sense, reasoning, and rationality*.

Russell, S. and Norvig, P. (2003). *Artificial Intelligence, A Modern Approach (2ème édition)*. Prentice Hall.

Schwartz, J. and Sharir, M. (1983). On the piano movers problem:I, II, III, IV, V. Technical report, New York University, Courant Institute, Department of Computer Sciences.

Simov, B., LaValle, S., and Slutzki, G. (2002). A complete pursuit-evasion algorithm for two pursuers using beam detection. In *Int. Conf. on Robotics and Automation (ICRA'02)*.

Svestka, P. (1997). *Robot Motion Planning using Probabilistic Roadmaps*. PhD thesis, Utrecht University.

Tovar, B., LaValle, S., and Murrieta, R. (2003). Optimal navigation and object finding without geometric maps or localization. In *Int. Conf. on Robotics and Automation (ICRA'03)*.

Williams, B. C., B.C., Kim, P., Hofbaur, M., How, J., Kennell, J., Loy, J., Ragno, R., Stedl, J., and Walcott, A. Model-based reactive programming of cooperative vehicles for mars exploration. In *Int. Symp. on Artificial Intelligence, Robotics and Automation in Space*, page 2001.

Yershova, A., Jaillet, L., Simeon, T., and LaValle, S. M. (2005). Dynamic-domain rrts: Efficient exploration by controlling the sampling domain. In *Int. Conf. on Robotics and Automation (ICRA'05)*.

TARGET VALUE PREDICTION FOR ONLINE OPTIMIZATION AT ENGINE TEST BEDS

Alexander Sung, Andreas Zell
Wilhelm-Schickard-Institute for Computer Science, University of Tübingen, Sand 1, Tübingen, Germany
Alexander.Sung@uni-tuebingen.de, Andreas.Zell@uni-tuebingen.de

Florian Klöpper, Alexander Vogel
Powertrain / Testing Methods, BMW Group, Munich, Germany
Florian.Kloepper@bmw.de, Alexander.Vogel@bmw.de

Keywords: Online Optimization, Engine Test Bed.

Abstract: The settling times of target functions play an important role in the domain of online optimization at the engine test bed. Inert target functions generally induce long measuring times which lead to increased costs. In this article, we analyze how previous knowledge about the physical behavior of target functions can be used to early predict the final steady state value to reduce measuring times.

1 INTRODUCTION

In recent years, model based algorithms have gained significance in the domain of online optimization of combustion engines (Isermann, 2003). In this area, known optimization systems like, for example, Cameo (Gschweitl et al., 2001) and Vega (Bredenbeck, 1999) are in use, but the algorithm *mbminimize* presented in (Knödler et al., 2003) (Knödler, 2004) (Poland et al., 2003) also has found its place in real applications.

An important and time consuming part of online optimization is the extraction of measuring data at the engine test bed. Inert target functions, like e.g. the exhaust gas temperature, need a long time until the final steady state value of the target function – also called target value – is reached for a certain combination of input parameters. In addition, the detection of the final value is complicated by noise. On the other hand, the behavior of transient oscillation is often known, or can be derived from physical rules in order to early predict the target value. This procedure is investigated in the given contribution, first in simulation and then on real engine data. Preceding analysis like (Flohr, 2005) (Schropp, 2006) show that the behavior of target functions in this application domain can often be described by simple mathematical functions. A target value prediction during the online optimization based on a small amount of data therefore has the potential to reduce the effort of measuring. The idea of prediction is not new, of course. There is recent work like (Castillo and Melin, 2002) (Han et al., 2004) (Teo et al., 2001) (Wang and Fu, 2005) which deals with this topic using approximators such as neural networks, genetic algorithms, or support vector machines for predictions based on time series.

2 TARGET VALUE PREDICTION

In this section, the algorithm of target value prediction is presented on idealized, artificially composed measuring data. In addition, the algorithm is tested with regard to robustness against noise.

2.1 The Principle of Target Value Prediction

The progression of target functions is often inversely exponential. A typical example is the temperature. At the engine test bed, the exhaust gas temperature is a quantity, which reacts slowly to adjustments of the input parameters in comparison with other quantities of the engine. It is thereby an ideal candicate for the following analyses for two reasons: On the one hand, it is easy to get a large amount of measuring data during the time of transient oscillation due to the inertia of the target function. On the other hand, the possible

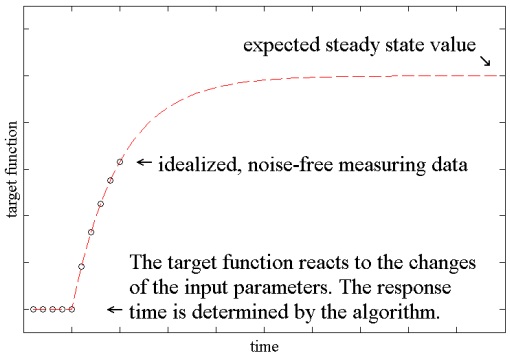


Figure 1: Idealized measuring data and the result of target value prediction.

profit of time gained by using target value prediction is especially large.

Figure 1 shows artificially generated, idealized measuring data of a target function with such a behavior, displayed by the black circles. The system is steady at the beginning and reacts with a delay to the adjustments of the input values. The recorded data is used as basis for online calculations by the target value prediction. With a function approximation algorithm based on least squares optimization, the behavior of the target function is emulated, and a target value is predicted. So, if the data is given as $\left(x_i, \tilde{f}(x_i)\right)$ with $\tilde{f}$ being the unknown target function, the objective is to find parameter values $\vec{v} = (v_1, v_2, v_3)$ that approximate the error

$$E = \sum_i \left(v_1 \cdot e^{v_2 \cdot x_i} + v_3 - \tilde{f}(x_i)\right)^2 \;\rightarrow\; \min! \qquad (1)$$

best. The result for the idealized data is also shown in figure 1. If the data is so ideal and free of noise like this, an early determination of the target value is simple. A greater challenge is given by the addition of noise, which is shown in the next section.

An important aspect of the prediction lies in the correct detection of the moment when the target function reacts to changes of the input. This point in time is unknown in a real application, and the response time often varies. In our applications, this moment is determinated by a significant change in the target values over a longer span of time. Therefore, it is fundamental for our algorithm to have a continuous and detailed record of the target function. Given the data $\left(x_i, \tilde{f}(x_i)\right)$, we first filter our plateaus by replacing every set

$$\{x_{i-n}, \ldots, x_i\} \,|\, \forall j \in \{1, \ldots, n\} : \tilde{f}(x_{i-j}) = \tilde{f}(x_i)$$

with its final data point x_i discarding the rest, then we choose the smallest x_i from the set

$$\left\{ x_i \,\middle|\, |\tilde{f}(x_{i+\delta}) - \tilde{f}(x_i)| > \lambda\left(\max_i \tilde{f}(x_i) - \min_i \tilde{f}(x_i)\right) \right\}.$$

δ and λ are empirically specified parameters.

During the online operation, the target value prediction is updated constantly with additionally recorded data. The algorithm is suited for online use, because the calculation of the target value only requires a few milliseconds. Depending on the noise, the predicted target value often changes when new measuring data is added, and the prediction improves in quality with an increased amount of data. The quality of the target value prediction is compared to the standard method, which is still in use at the test beds at present. The standard method assumes that the target function is tuned after a fixed amount of waiting time, specified in advance. Then, an averaged measuring value is used as final target value. In our simulation the waiting time, also referred to as stationary time, is set to the point in time when the target function has reached 95% of its final value. At this point, a measuring time of half the stationary time's duration is used. This is in accordance to the situation at the engine test bed.

2.2 The Inpact of Noise

To test the robustness of the algorithm against noise, the following tests were done. For different noise levels 1000 test runs were evaluated each. Normal distributed noise was used with a standard deviation specified in relation to the range of the target values. The results are the averaged saving of time in comparison to the standard method with a fixed stationary time, as well as the relative error reduction ε, calculated as the fraction of error ε_{Ref} of the standard method by which the prediction error ε_{Pred} is lower, and vice versa:

$$\varepsilon = \begin{cases} \frac{\varepsilon_{Ref} - \varepsilon_{Pred}}{\varepsilon_{Ref}} & \text{if } \varepsilon_{Pred} < \varepsilon_{Ref} \\ \frac{\varepsilon_{Ref} - \varepsilon_{Pred}}{\varepsilon_{Pred}} & \text{otherwise} \end{cases}. \qquad (2)$$

The higher the values are, the better target value prediction worked. The prediction success displays how often the prediction error was smaller than the error of the standard method. The results are presented in table 1.

Table 1: Investigation of noise influence.

Noise level	Saving of time	Error reduction	Prediction success
1%	46.1%	68.4%	96.6%
2%	42.7%	42.6%	80.1%
3%	39.3%	26.6%	68.4%
5%	33.9%	7.5%	55.9%
10%	23.2%	0.0%	46.6%

By means of these results, it was possible to confirm the assumption that the function approximation

method of the target value prediction is able to determine the final value more precisely than a simple averaging after a fixed stationary time. With increased noise, which significantly falsifies the measuring data especially in the early phase of transient oscillation, the difference lessens because an early prediction is not possible anymore in this case.

At a high noise level, which may commonly occur in real applications, an early prediction is still possible, but the success is lacking. In the next section, we show how this trade-off can be used to reduce the overall measuring time of an optimization problem without downgrading the final results.

Figure 2 shows sample data sets with different noise levels.

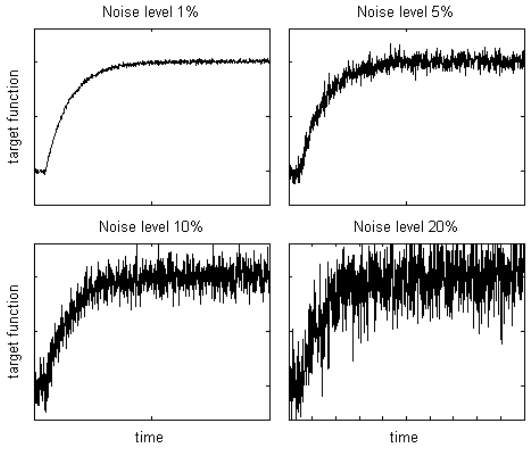


Figure 2: Sample data with different noise levels.

2.3 Addressing More Than Simple E-functions

The simple E-function is only one of different possibilities to describe the behavior of target functions. In our real application, for example, we found that a twofold E-function $v_1 \cdot e^{v_2 \cdot x_i} + v_3 \cdot e^{v_4 \cdot x_i} + v_5$ was better suited to describe the progression of the measuring data. This and other unexpected behavior in the data may be explained by the measuring technology, the interplay of which still needs more investigations. In this contribution, we want to mention two examples, which occur commonly at the engine test bed.

Figure 3 shows an example of real engine data with an overshooting behavior. In this case, the target values react to changes of the input parameters by progressing past the final steady state value first, and then continue retrogressive afterwards. The difficulty lies in the fact that the overshooting behavior cannot be detected before the retrogressive part is recorded.

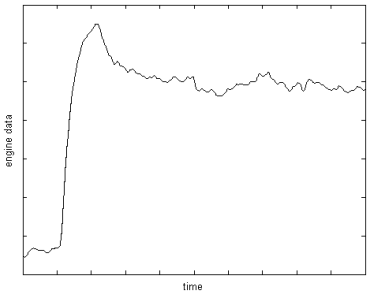


Figure 3: An example of real engine data with overshooting behavior.

An early target value prediction will therefore fail to describe the progession correctly. For an optimization problem with several measuring points, the decision whether to abort the measuring process due to an early target value prediction can be made depending on the significance of the expected target value. For example, if the current target value is already under a certain threshold of importance and the predicted value has even less impact on the optimization result, a possible overshooting can be neglected.

Sometimes, a drift in the target values can be detected. This behavior can be identified with target value predictions, too. If several predictions are recorded over a longer period of time, it becomes evident that a small linear time is included in the underlying target function $v_1 \cdot e^{v_2 \cdot x_i} + v_3 \cdot x_i + v_4$. In general, however, the influence of drift at one single measuring point is too small to have a significant impact on the final steady state value. Since the detection of drift behavior is complicated by noise and requires a longer period of measuring time, we neglect the possible occurrence of drift in general. This corresponds to the standard measuring method, which determines the final result as an averaged measuring value.

3 RESULTS IN REAL APPLICATIONS

This section consists of several applications from the domains of both simulation and the engine test bed, in which target value prediction improves the final results.

3.1 Model-based Optimization

In the applications of this section, we used a model-based approach for the optimization of the input parameters of a target function that was defined over a given input space. The models of the target function were generated by a committee of neural networks

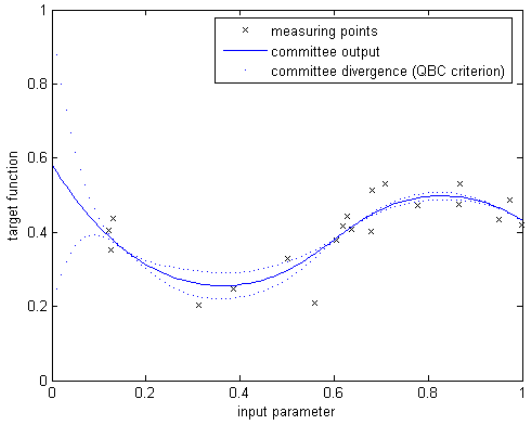


Figure 4: Model output of a committee of neural networks and linear regression. The QBC criterion is a measurement for the uncertainty and supplies the largest values to the points where the committee's individual models diverge the most.

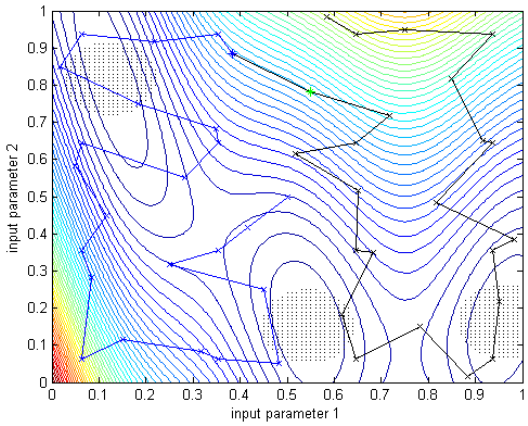


Figure 5: Snapshot of the online optimization in a simulation environment. The black dots mark the test error set.

and LLR models, the output of the later being a linear combination of non-linear basic functions. Neural networks have been long established in model-based optimization, as illustrated in (Hafner, 2002) (Hafner et al., 2000) (Schüler et al., 2000), for example. The strengths of neural networks for our use lies in the ability to produce a good overall approximation of the target function based on few data points without prior knowledge. The LLR models, on the other hand, are designed to specifically describe local behavior in the input space in detail.

Based on such a model, we calculate a QBC criterion (query by committee), of which figure 4 provides an example over an one-dimensional input space. The QBC criterion supplies the largest values to the points where the committee's individual models diverge the most. The information of already known measuring points is included to avoid repeated measuring at the same input values. The expression "query" here denotes the decision-making process to determine the next measuring point with the aid of the currently available model. This method will be used in the following subsection.

3.2 Optimization in a Simulation Environment

As a first application, we used our algorithm in a simulation environment, which was designed for the optimization of an engine model. Measuring points were chosen in a two-dimensional search space by a D-optimal DoE-method (Röpke, 2005) (Weber et al., 2005). The goal was to find the local minima and to model the regions around them with the least error possible. We used the Branin function as the target function and simulated it to be inert, according to what was described in the section before. Note, that the test set consisted of test points near the local minima only, in order to display the optimization goal. Figure 5 shows the setup of the experiments. Our algorithm was tested against the standard method with fixed stationary and measuring time.

There are two conditions, after which the prediction based algorithm completes the measuring process at a specific measuring point and evaluates the gathered measuring data via prediction: In the standard case, analogous to the section before, the measuring process is finished when the predicted value is stable over a certain period of time. This is calculated with the term

$$\begin{aligned} &\left| mean\big(p(t-\delta),\ldots,p(t-1)\big) - p(t) \right| < \alpha \\ &\quad \wedge \quad std\big(p(t-\delta),\ldots,p(t-1)\big) < \alpha \end{aligned} \tag{3}$$

using the predicted values $p(t)$ after normalization. The parameters $\delta = 30$ (seconds) and $\alpha = 0.01$ are empirically determined based on real engine data.

In addition, however, there is a possibility to abort the measuring when the expected target value is too high to be a local minimum. In this case, the value predicted at that moment is used as an estimated target value, even if it is to be expected that the error at this point is quite large. The intention of this method is to save measuring time at unimportant measuring points to gain a first rough engine model und use the saved time to explore the regions of minima in detail. Figure 6 shows two cases where the measuring process was completed due to early predictions. In the first case, the measuring was complete since the predicted values leveled off at the point in time marked by the dotted line. The second case shows an example where the expected value was considered irrelevant for the minimization problem. The measuring process was aborted in this case.

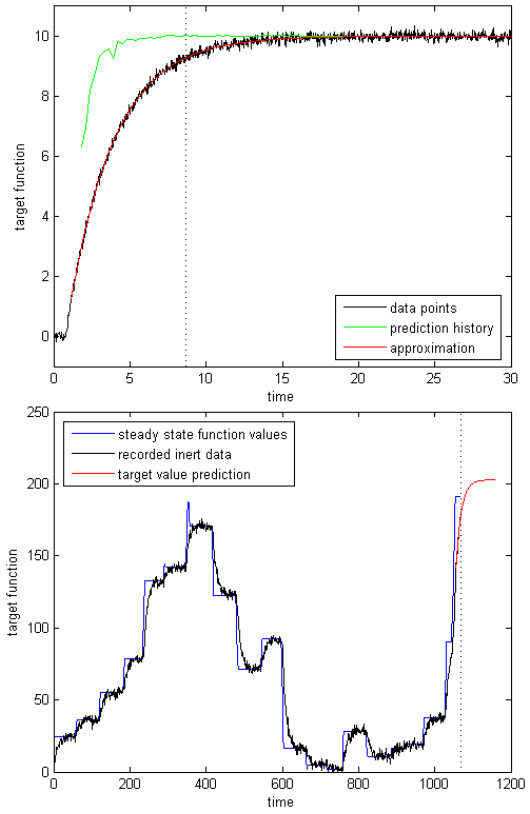


Figure 6: Examples of measuring processes completed due to the evaluations of predicted values. In the first example, the prediction is considered reliable, in the second example, the expected value is considered umimportant for the optimization.

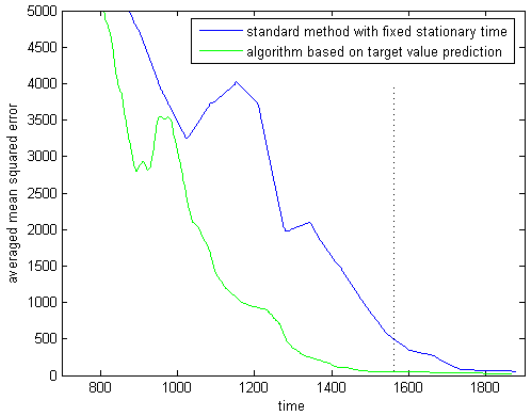


Figure 7: Test error progression of the online optimization in a simulation environment. The standard method is displayed in blue, the target value prediction algorithm is shown in green. The dotted line shows when the target value prediction algorithm completed the initial set of data points.

Two additional rules proved to be useful for the optimization. Measurings with target values near the expected optimum will not be aborted, and measurings near local minima based on the current, incomplete model will also not be aborted. The thresholds for these options are set by the user based on rough previous knowledge at this time, but they are not sensitive. These rules grant an additional certainty that no information will be lost unnecessarily.

Figure 7 shows the progression of the test error during the optimization over time, as an average over 72 test runs. After 74.4% of the time that the standard method needed in total, the target value prediction algorithm evaluated the initial set of measuring points. At that point, our algorithm created a function model with a test error of 41.6, while the model of the standard method still had a test error of 502.6. The reason for this huge difference lies in the fact that the standard method did not yet deal with several measuring points, so the error near those is still very large, of course. The final test error of the standard method was 31.4.

In addition to the initial set of measuring points, the prediction based algorithm used the saved time to measure additional data points. To generate these points, the current model of the target function was used to locate the regions near local minima, as explained in the subsection before. Based on QBC criterion, the additional measuring points were chosen by a line search algorithm. With the additional data points, the final test error could be reduced to 17.7 with a 43.6% improvement in comparison with the standard method.

In this scenario, our algorithm was able to produce a rough interim result in a shorter time. On the other hand, using the complete amount of time available, our algorithm was able to improve the final results in comparison with the standard method.

3.3 Target Value Prediction on Real Engine Data

As an example with real application data, the target value prediction algorithm was used on engine data, which was provided by the BMW Group in Munich. The data consists of exhaust gas measurings and describe load step responses. The range of values has been scaled to make the relative error values comparable. The results contain 126 data sets.

Dealing with real engine data poses a problem, which makes a presentation of the results according to simulation difficult: The real final values of the target function are unknown. Even after measuring for minutes, it is possible that the transient oscillation of several target functions still have not finished. An open-ended measuring is not viable, however. Figure

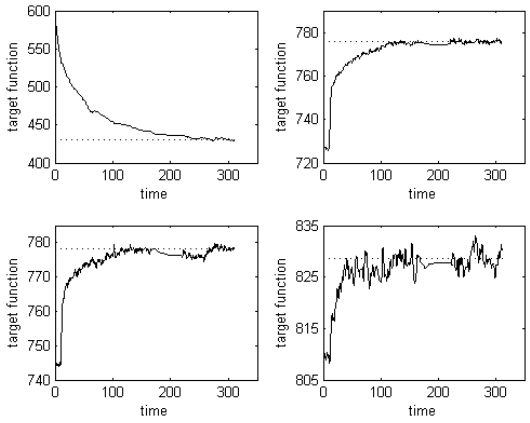


Figure 8: Examples of recorded engine data, which still have not reached their final values at the end of the recording. The plateaus are a phenomenon of the measuring technology at the engine test bed. For comparison, the predicted target value is included as a dotted line.

8 shows several examples of recorded engine data, which still have not reached their final values at the end of the recording. Plateaus in the recorded data arise because the measuring process is disabled during the time of evaluating and saving the engine data at the engine test bed. The data points after this point in time still vary, however. Therefore, additional assumptions, which are derived from the simulation, are made for the following data evaluations.

In the practice of test bed operations, target functions are assumed to have reached their final values after a certain amount of stationary time. A concluding averaged value is then accepted as target value. In the given data, the stationary time has been set to 60 seconds. Afterwards, a measuring has been performed over a timespan of 30 seconds, after which the averaged value has been calculated. This value was used for comparisons to rate the target value prediction algorithm. The results from simulation show that with an appropriate modeling of the target function progression given, the target value can be estimated more precisely by a corresponding prediction than by an averaged measuring value. For this reason we assume that, in the case of real engine data, the predicted target value, which was calculated based on all available measuring data, approximates the real final value best. The examples in figure 8 include the final predicted target value for comparison. With regard to this value, the results can be illustrated as in table 2.

In the first column, a maximum error level is given. The second column shows in how many cases this error level was reached. The error level was reached when the difference of the prediction result to the expected final value did not exceed the given

Table 2: Evaluation of error levels using target value prediction.

Maximum error	Threshold success	Saving of time
3%	14.3%	21.4%
5%	27.0%	25.1%
10%	63.5%	38.4%

level for at least 10 seconds. The difference is again calculated in relation to the range of the target function. In the last column, the averaged amount of time is shown that could be saved in those cases where the error level was reached. The results for the comparison with the standard method in analogy to simulation are a saving of time of 26.4% and an error reduction of 3.3%. The prediction thereby gave a better result in 52.4% of the tests.

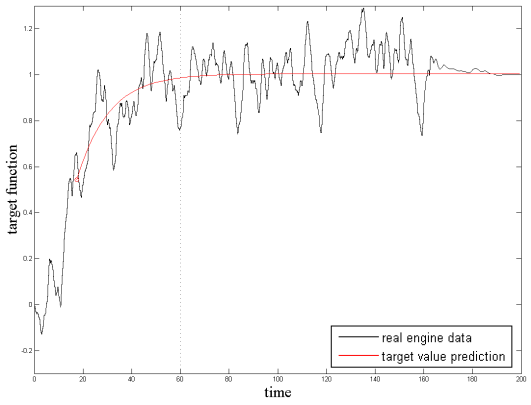


Figure 9: Successful target value prediction (red line) applied on real engine data (black line). Only the first part on the left side of the dotted line was used for the prediction. The real target value progression is shown for comparison.

Figure 9 shows an exemplary and successful target value prediction based on engine data. In this case, a similarity to the simulated E-function is identificable. The complete engine data is drawn in black, while the green dashed line highlights what data was available to the prediction algorithm. The predicted behavior is shown in red.

3.4 Model-based Optimization of a Combustion Engine

As another application, we modeled a target function of an engine over a two-dimensional parameter space. The problem we had hereby was that there was not enough real engine data available for an additional large, independet test set. So, to evaluate the algorithms on the real application, we had to create a simulated reference. We did this by creating an engine

model with the given engine data. This served as a basis for calculations about test errors.

In analogy to the subsection before, we wanted to explore the input space by measuring certain data points. The target values were real engine data, and therefore the measuring process was inert. With the acquired data, a model was created both for our algorithm based on target value prediction as well as for the standard method using stationary and measuring times. The goal was to find the local minima with a good representation of their immediate environments. Note that we are refering to three different models now. The first one is created with steady state values and serves as reference. The other two models are calculated during the optimization process where recorded engine data on the inert target function is evaluated. Figure 10 shows the first of these models which represents the engine's target function in steady states. Another fourth model is created by using target value prediction without the goal to minimize optimization time, but to achieve best test error values instead.

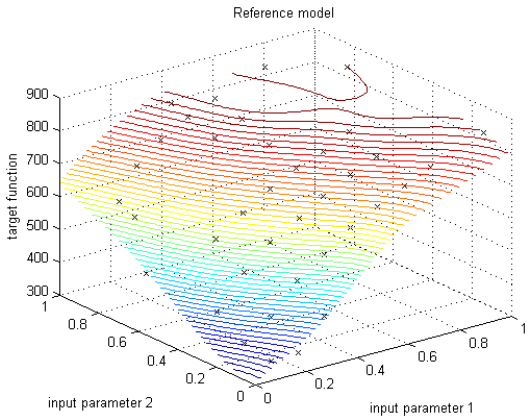


Figure 10: Model of the target function based on real engine data, and the real measuring data. The function values from this model are used as reference in our tests.

The measuring process consisted of approaching and measuring 38 data points. Using the standard method, the final mean squared error was 231.1. The algorithm using target value prediction was able to reach an error of 147.4 after 75.2% of the time which the standard method needed, an improvement of 36.8% in comparison to the standard method. Therefore, the target value prediction could be used to save time to early generate an engine model that had a lower test error based on the reference model. Furthermore, the predictions can also be done using the same amount of time which the standard method uses. In this variation the final result was 139.3, an improvement of 39.7%.

4 CONCLUSION AND PROSPECTS

In this article we described the idea and methodology of target value prediction. Thereby, the fundamental assumption was that the behavior of the target function after adjusting the input parameters can be described with inversely exponential E-functions. Results from both simulation and practice show the possible success of this algorithm. We demonstrated that the strength of the target value prediction does not lie in single measurings, since there is always some sort of trade-off between saving of time and precision loss included. Given a larger scope of an entire optimization problem, however, exactly this trade-off can be used to concentrate available resources to the important parts of the problem and to save valuable time at less significant aspects.

Due to the nature of the prediction method to detect the important regions of an optimization problem, we expect the algorithm to scale well with larger problems where the areas of solutions do not scale accordingly at all. Further work will apply the proposed method to other problems in the domain of online optimization and continue to show its capacity.

REFERENCES

Bredenbeck, J. (1999). Statistische Versuchsplanung für die Online-Optimierung von Verbrennungsmotoren. In *MTZ-Motortechnische Zeitschrift und MTZ-Worldwide, 60 (11)*. MTZ Press.

Castillo, O. and Melin, P. (2002). Hybrid intelligent systems for time series prediction using neural networks, fuzzy logic, and fractal theory. In *IEEE Transactions on Neural Networks, 13 no. 6*, pages 1395–1408.

Flohr, A. (2005). *Konzept und Umsetzung einer Online-Messdatendiagnose an Motorprüfständen*. PhD thesis, University of Technology Darmstadt.

Gschweitl, K., Pfluegl, H., Fortuna, T., and Leithgoeb, R. (2001). Steigerung der Effizienz in der modellbasierten Motorenapplikation durch die neue CAMEO Online DoE-Toolbox. In *ATZ-Automobiltechnische Zeitschrift und ATZ-Worldwide*, page 103 (7/8). MTZ Press.

Hafner, M. (2002). *Modellbasierte stationäre und dynamische Optimierung von Verbrennungsmotoren am Motorenprüfstand unter Verwendung neuronaler Netze*. PhD thesis, University of Technology Darmstadt.

Hafner, M., Schüler, M., and Isermann, R. (2000). Einsatz schneller neuronaler Netze zur modellbasierten Optimierung von Verbrennungsmotoren - Teil 2: Stationäre und dynamische Optimierung von Verbrauch und Emissionen. In *Motortechnische Zeitschrift (MTZ), 61 no. 11*. MTZ Press.

Han, M., Xi, J., Xu, S., and Yin, F.-L. (2004). Prediction of chaotic time series based on the recurrent predictor neural network. In *IEEE Transactions on Signal Processing, 52 no. 12*, pages 3409–3416.

Isermann, R. (2003). *Modellgestützte Steuerung, Regelung und Diagnose von Verbrennungsmotoren*. Springer-Verlag.

Knödler, K. (2004). *Methoden der restringierten Online-Optimierung zur Basisapplikation moderner Verbrennungsmotoren*. PhD thesis, University of Tübingen.

Knödler, K., Poland, J., Zell, A., Fleischhauer, T., Mitterer, A., and Ullmann, S. (2003). Model-based online optimization of modern internal combustion engines, part 2: Limits of the feasible search space. In *MTZ Worldwide (Motortechnische Zeitschrift), 64 no. 6*, pages 30–32, German edition 520–526. MTZ Press.

Poland, J., Knödler, K., Zell, A., Fleischhauer, T., Mitterer, A., and Ullmann, S. (2003). Model-based online optimization of modern internal combustion engines, part 1: Active learning. In *MTZ Worldwide (Motortechnische Zeitschrift), 64 no. 5*, pages 31–33, (German edition: 432–437). MTZ Press.

Röpke, K. (2005). DoE – Design of Experiments. In *Methoden und Anwendungen in der Motorenentwicklung*. verlag moderne industrie.

Schropp, S. (2006). Optimization of measurement data logging at engine test benches. Master's thesis, University of Technology Munich.

Schüler, M., Hafner, M., and Isermann, R. (2000). Einsatz schneller neuronaler Netze zur modellbasierten Optimierung von Verbrennungsmotoren - Teil 1: Modellbildung des Motor- und Abgasverhaltens. In *Motortechnische Zeitschrift (MTZ), 61 no. 10*. MTZ Press.

Teo, K. K., Wang, L., and Lin, Z. (2001). Wavelet packet multi-layer perceptron for chaotic time series prediction: Effects of weight initialization. In *Lecture Notes in Computer Science, Vol. 2074*, pages 310–317.

Wang, L. and Fu, X. (2005). Data mining with computational intelligence. In *IEEE Transactions on Neural Networks, 17 no. 3*. Springer.

Weber, M., Kötter, H., Schreiber, A., and Isermann, R. (2005). Model-based Design of Experiments for Static and Dynamic Measurements of Combustion Engines. In *3. Tagung: Design of Experiments (DoE) in der Motorenentwicklung, Berlin*.

SHORT PAPERS

A NEURAL-CONTROL SYSTEM FOR A HUMANOID ARTIFICIAL ARM

Michele Folgheraiter, Giuseppina Gini and Massimo Cavallari
Department of Electronics and Information, Politecnico di Milano, Piazza L. da Vinci 32, Milano, I-20133, Italy
folghera@elet.polimi.it, gini@elet.polimi.it, massimocav@elitel.biz

Keywords: Biorobotics, Artificial Arm, Neural Controller, Humanoids Robotics.

Abstract: In this paper we illustrate the architecture and the main features of a bio-inspired control system employed to govern an anthropomorphic artificial Arm. The manipulation system we developed was designed starting from a deeply study of the human limb from the anatomical, physiological and neurological point of view. In accordance with the general view of the Biorobotics field we try to replicate the structure and the functionalities of the natural limb. Thanks to this biomimetic approach we obtained a system that can perform movements similar to those of the natural limb.
The control system is organized in a hierarchical way. The low level controller emulates the neural circuits located in the human spinal cord and is charged to reproduce the reflexes behaviors and to control the arm stiffness. The high level control system generates the arm trajectory performing the inverse kinematics and furnishing the instantaneous muscles reference position. In particular we implemented the Inverse kinematic using a gradient based algorithm; at each step the actuators movements are arranged in order to decrease the distance between the wrist and the target position.
Simulation and experimental results shows the ability of the control system in governing the arm to follow a predefined trajectory and to perform human like reflexes behaviors.

1 INTRODUCTION

Robotics since from the beginning was involved in replicating the human manipulation capabilities. One of the first manipulators, designed for research purposes, was the Stanford Arm (Stanford Artificial Intelligence Laboratory). This robot was furnished of 6 DOFs (Degree Of Freedom), five revolute joints and one prismatic. Even if the mechanical architecture was thought with the aim to emulate the human movements we can not consider it as a real anthropomorphic system.

In sixties General Motor (the first to employ a Manipulator in an industrial process) financed a research program at MIT to developed another well know robotics arm: the PUMA (Programmable Universal Manipulator for assembly).
We can classify this manipulator as anthropomorphic, indeed it is possible to compare it to a human arm. At first we can divide the mechanical structure in three principal blocks: the shoulder with two DOF, the elbow with 1 DOF and the wrist with another three DOF. The Puma has a dexterity that is quite near to that of a human arm, even though the human shoulder has more than two DOF. The analogy between the human arm and the PUMA manipulator is true only from a kinematic point of view, because the two systems have completely different performances. We can assert that this robot is more precise than a human arm, nevertheless the natural arm is structurally conformed to perform tasks that require compliant features, like clean an irregular surface or catch a ball.

A fine regulation of the joint stiffness is therefore very important also in a Humanoid Robot, indeed if we want that the robot will be able to perform a task in collaboration with a human being we have the necessity to change the joints stiffness in concomitance with the robot movements. In industrial manipulators it is possible to set the end-effector compliance using the information coming from a three axis force sensor

installed on the wrist, nevertheless this is quite similar to a virtual compliance control. In fact, the motors that equip the manipulator do not present an intrinsical compliance, normally if the power is switch off the robot assumes its maximum rigidity. This behavior is not acceptable for a robot that is thought to operate in strict contact with a human being.

With this work we want to propose a different methodology in developing such a systems, not only we tried to emulate the human arm morphology, but we implemented also a neural controller that replicates the functionalities of primary motor circuits located in the human spinal cord. We are convinced that combining classical control methodologies with bio-inspired one may brings a new class of machines that will perform better.

There are many projects that attempted to design an arm with human like features and capabilities. At the Center for Intelligent Systems (Vanderbilt University) Prof. Kawamura and its group are working on the ISAC humanoid robot. This robot consists of a human-like trunk equipped with two six-DOF arms moved by McKibben artificial muscles (Kawamura et al., 2000). Each joint is actuated by two antagonistic actuators that are controlled by a system able to emulate the electromyogram patterns (EMG) of a human muscle. The arm, during a fast reaching movement, can avoid an obstacle performing a reflex behavior (Gamal et al., 1992), furthermore the phasic pattern is autonomously adjusted when a reach trajectory doesn't closely match a desired response. The main advantage of this bio-mimetic control architecture is the possibility to reduce the joint stiffness during a movement execution.
Another project in the same direction is that one at the Biorobotics Laboratory in Washington University. Here Prof. Hannaford and his team have worked intensely on the emulation of the human arm (Hannaford and Chou, 1997) (Hannaford et al., 1995). The goal of this research is to transfer knowledge from human neuro-musculo-skeletal motion control to robotics in order to design an "anthroform" robotic arm system.

Following the bio-mimetic approach they developed and tested new kind of actuators (Klute and Hannaford, 2000) and sensors (Hannaford et al., 2001) (Jaax, 2001), whose purpose is to replicate a mammalian muscle spindle cell, that measures the contraction and the muscle velocity.

Another very interesting arm project is that one developed by Department of Precision Machinery Engineering in University of Tokyo. The system(Toshiki Niino and Higuchi, 1994) is equipped with a new types of compact, high-power, electrostatically driven actuators.

The actuators have 40 or 50 pairs of sheets interleaved together and enclosed in ready-to-use packages filled with a dielectric liquid. An electrostatic artificial muscle consists of two groups of sheets stacked and interleaved together. Sliding forces are generated on the surface of each film and are combined into a net force at the bundled edges of the sheets.

A first type of pulse-drive induction artificial muscle, which utilizes induced charges on highly resistive films, generated 8N propulsive force and 0.5W mechanical power with 110g its own mass. At the Dept. of Mechanical Engineering (Vrije Universiteit Brussel) Prof. Frank Daerden et al designed a new type of Pneumatic Artificial Muscle (PMA), namely the Pleated Pneumatic Artificial Muscle (PPAM) (Daerden and Lefeber, 2001). It was developed as an improvement with regard to existing types of PAM, e.g. the McKibben muscle (Klute and Hannaford, 2000).

Its principal characteristic is its pleated membrane. It can inflate without material stretching and friction and has practically no stress in the direction perpendicular to its axis of symmetry. Besides these it is extremely strong and yet very lightweight and it has a large stroke compared to other designs.

2 THE ARM ARCHITECTURE AND FEATURES

In 2003 we developed a first prototype of artificial Arm (MaximumOne Figure 1) with the main aim to experience the actuation architecture and the control strategies we theorized.

From the kinematic point of view, if we exclude the hand, the system presents overall four degrees of freedom. Three are located in the shoulder that resembles a spherical joint, and one in the elbow that is a normal revolution joint.

Each joint is actuated by tendons connected with McKibben artificial muscles. In order to detect the arm posture and allow to close the control loop, each actuator is equipped with a position and force sensor. These devices were developed specifically for this application, this because there are not commercial system that meet our needs. Furthermore the elbow joint is furnished of an angular sensor (Figure 1) that measures the joint position.

The signals coming from the sensors are conditioned and gathered by dedicated boards that perform an analog multiplexing and send the information to a **Target-PC** equipped with a real-time kernel. The control system, implemented via software

on a **Host-PC**, receives and elaborates these information via RS-232 serial connection and send back its output, through the Target-PC, to the electro-valves module that has the main purpose to set the actuators pressures. This module is equipped with 14 micro solenoid-valves that can operate at a maximum frequency of 20Hz. Using a PWM (Pulse Wide Modulation) modulation it is possible to regulate the air flow that feeds the single actuator and therefore its force and position.

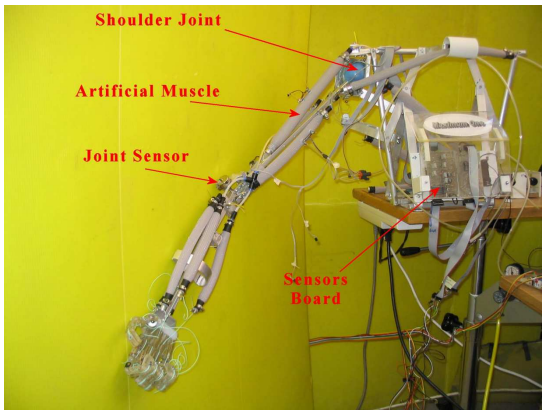


Figure 1: MaximumOne,the Arm Prototype of the Artificial Intelligence and Robotics Laboratory, Politecnico di Milano.

As it is possible to see from the picture (Figure 1), the arm we developed has an anthropomorphic shape. This because during the design, we have tried to reproduce the human arm dimensions and proportions, the articulation mobilities, the muscle structure, and the same sensorial capabilities.

2.1 The Actuation System

In order to actuate the arm we used seven artificial muscles (Figure 2): five are dedicated to the shoulder joint and two to actuate the elbow. This permits us to obtain the typical postures and movements of the natural limb. The five shoulder actuators emulate the mechanical functionalities of: pectoralis major, dorsal major, deltoid, supraspinatus and subscapularis muscles. The two elbow actuators emulate the functions of biceps and triceps muscles.

If we compare the the human arm musculature with the actuation system of our prototype we can find out same differences; for example the actuators that represent the biceps and triceps muscles are monolink in the sense that they are dedicated only for the elbow actuation, instead in the human limb they interest at the same time the shoulder and elbow articulations. Furthermore shoulder actuators are placed

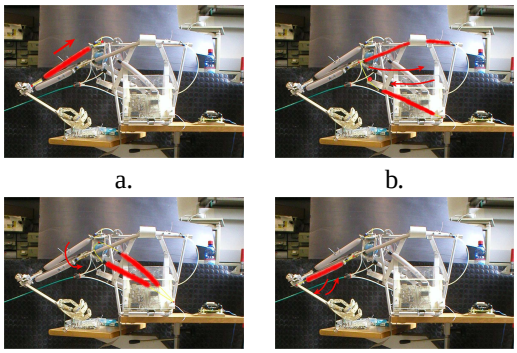


Figure 2: (a) The Deltoid Actuator lifts the shoulder. (b)The Pectoral and Dorsal Actuators allow the adduction and abduction movements. (C)The Supraspinatus and Subscapularis Actuators allow the shoulder rotation. (d) The Biceps and Triceps Actuators allow the elbow flexion and extension.

in a manner to maximize the space available for their movement. Indeed, because of McKibben actuators can contract only the 20% of their maximum length, depending on the tendon excursion we need to realize, their minimum length is fixed.

3 CONFIGURATION OF THE CONTROL SYSTEM

The control system of the arm is organized in a modular and hierarchical fashion. At the bottom level (Figure 3) there are the artificial reflex modules(Michele Folgheraiter, 2004) that govern the actuator's contraction and force. These modules receive inputs from the joint path generator, which in turn is fed by the Inverse Kinematic module that computes the target actuators lengths. The inputs of the entire control system are: the final hand position in the cartesian space, and the P signal that scales the level of artificial muscles co-activation (simultaneously activation of the muscle that govern the same joint).

From a hierarchical point of view, we can distinguish three main levels:

- **High level controller**: composed by the Inverse Kinematic modules
- **Medium level controller**: composed by the path generator module
- **Low level controller**: composed of the reflex modules that control the artificial muscles activities

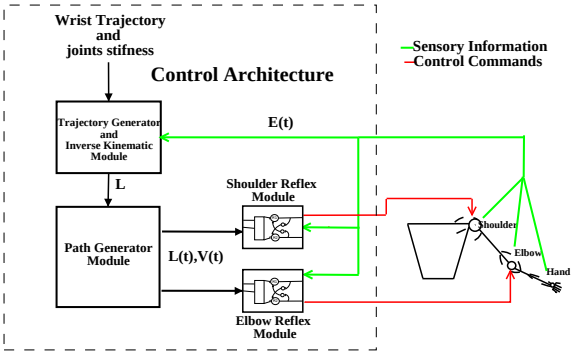


Figure 3: Control System Architecture.

The signals transmitted from one module to another are expressed in a vectorial form, where each vector component corresponds to one of the seven artificial muscles that compose the actuation system. Therefore L_T represents the target lengths vector for the actuators, V_T represents the target velocity vector, E_L represents the length vector error, and P is the stiffness command vector. At the level of each single module these signals are decomposed in their components and sent to the appropriate submodules.

4 ARTIFICIAL NEURAL CIRCUITS TO IMPLEMENT THE REFLEXES BEHAVIORS

Reflex behaviors are accomplished by two modules that implement a simplified model of the natural circuits present in the human spinal cord. One module is dedicated to the control of the artificial muscles activities that govern the shoulder joint, the other takes under control muscles that actuate the elbow joint. Since the artificial muscle is constituted by only one functional fiber the biological organization of the natural muscle in motor units is neglected in our model. The artificial muscle activity is therefore regulated by only one motoneuron. The same consideration can be done also for the sensorial system in the muscle, that in this case is constituted by only one artificial spindle organ and only one artificial Golgi tendon organ.

The most important cells of the neural circuit are the motoneurons whose outputs set the actuators pressures. With respect to other models in literature (Sterratt, 2001),(Kuntimad and Ranganath, 1999),(Folgheraiter and Gini, 2001) or to hardware solutions (Omura, 1999) we decided to neglect the spike behavior of these cells, instead we concentrated our attention on modelling its membrane potential.

Each motoneuron receives its inputs from almost all the cells that compose the circuit. In equation 1 M_i represents the potential (membrane potential) of the motoneuron **i**.

$$\frac{d}{dt}M_i = (1 - M_i)(exc_i) - M_i(inh_i) \qquad (1)$$

where the terms exc_i and inh_i are the excitatory and inhibitory inputs for the motoneuron.

The neuron output is represented by equation 2

$$Mo_i = Th(M_i) \qquad (2)$$

where the threshold function is defined by equations 3 :

$$Th(x) = \begin{cases} x & \text{if } 0 \leq x \leq 1 \\ 0 & \text{if } x \leq 0 \\ 1 & \text{if } x \geq 1 \end{cases} \qquad (3)$$

For more details about this model see (blind).

4.1 Elbow Neural Circuit

The reflex module that governs the elbow muscle is represented in figure 4. It implements an opponent force controller whose purposes are to provide inputs for the path generator module, measure movements error and return error signals when the execution is different from the desired movement.
In figure 4 M_6 and M_7 are the motoneurons that control the contraction rate and force of the triceps and biceps actuators respectively. I_a6 and I_a7 are the interneurons that receive the error signals from the artificial spindles and project, with inhibitory synapses, to the motoneurons of the antagonist muscles M_7 and M_6 respectively. R_6 and R_7 represent the Renshaw cells that receive the error signals from spindles and inhibit the corresponding motoneuron and I_a cell, they are important to reduce oscillations of the joint around the target angular position. I_b6 and I_b7 are interneurons that receive the signals coming from the artificial Golgi tendon organs (that in this system are represented by a normalized force measurements). $I_{ns}6$ and $I_{ns}7$ are the interneurons that integrate information of stiffness and target length commands.
Finally M_s6 and M_s7 represent the artificial muscle spindle receptors. As inputs they receive the muscle velocity command, the muscle target length command and the actual muscle length and in turn excite the corresponding motoneuron and I_a interneurons.

In this work we do not consider the other inputs (show in the neural circuit schema) for more details see (Folgheraiter, 2003).

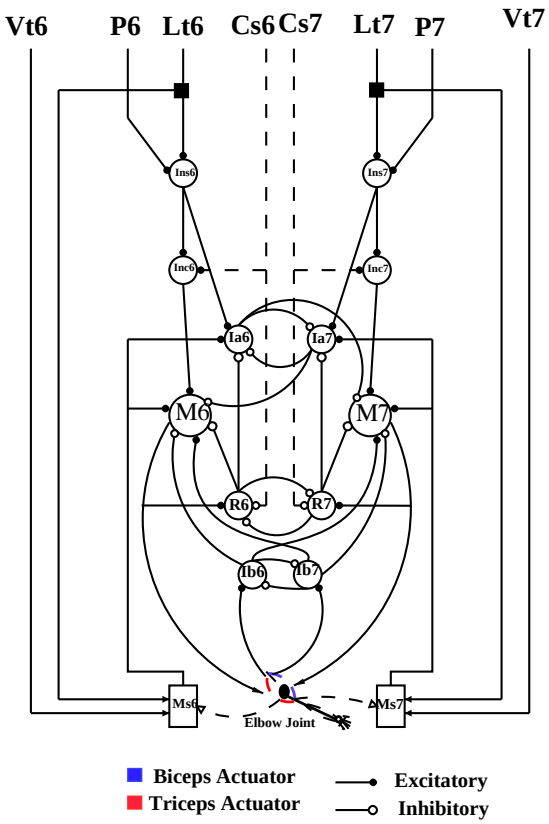


Figure 4: Architecture of the Elbow Reflex Module.

5 TRAJECTORY GENERATION: DIRECT AND INVERSE KINEMATIC

In order to test the efficacy of the control system we developed two kind of arm models:

- **The Kinematic Model**: it is used in the inverse kinematic algorithm to calculate the distance between the target position and the actual position for the wrist.
- **The Dynamic Model**: it represents a realistic model of the arm and takes into account the dynamic features of the actuators and of the arm links.

In this paper we will briefly describe only the Direct Kinematic model, because of it is required to develops the Arm Inverse Kinematic algorithm.

6 DIRECT KINEMATICS

Find the Direct Kinematics of the arm means to find the function that relates the actuator lengths vector with the wrist position. In this case the orientation is not considered because, if we do not take into account the hand, the kinematic chain, with only four DOF (three in the shoulder and one in the elbow), does not allows to set arbitrarily both the position and orientation of the wrist. It is possible to find the Direct Kinematic of the arm solving a system of equations, where each one imposes a constraint on the arm position. As an example, if we consider the pectoral and dorsal actuators we can obtains equation 4.

$$\begin{cases} (x-Opc_x)^2+(y-Opc_y)^2+(z-Opc_z)^2-Lpc^2=0 \\ (x-Odo_x)^2+(y-Odo_y)^2+(z-Odo_z)^2-Ldo^2=0 \\ (x-S_x)^2+(y-S_y)^2+(z-S_z)^2-(\|S-Opc\|)^2=0 \end{cases} \quad (4)$$

In equation (Eq. 4) (x,y,z) is the position where the two actuators are connected to the upper-arm, Opc and Odo represent the origins of the pectoralis and dorsal respectively, Lpc and Ldo are the actuator lengths, and S is the shoulder position. All the quantities are referred to the Arm reference system. The first two equations impose that the distance between the two extremities of the artificial muscle are equal to the required distance, instead the third equation imposes the condition that at the end of the movement the position of the attachment of the artificial muscles compared to the bone doesn't result modified. As it is possible to see equation 4 has two solutions and should be recalculated every time the position of the upper arm changes, indeed it depends on (x,y,z) a point located on the first robot link.

7 INVERSE KINEMATICS

Inverse Kinematics allows the calculation of the muscles lengths vectors when a target position in the robot workspace is assigned.

The algorithm we implemented for the Inverse Kinematics is based on the gradient descent method, this in order to calculate the minimum of the distance function between the current and the target wrist position. It approaches a local minimum by taking steps that are proportional to the negative of the gradient.
We can codify the algorithm using the pseudo-code of figure 5.

Where ε_{max} is the maximum error allowed in reaching the target position, and subsets $Sub(i)$ represents a couple of actuators chosen with a specific

```
- Set Current Wrist Position
- Set Target Wrist Position
- Calculate Distance between Current Position and  Target Position

while ( Distance > ε_max)
   for i=1:4
       for each muscle  belonging  Sub(i)
              - Decrease  its length
              - Calculate the effect  on the end-effector position  ( Direct Kinematic)
              - Calculate the  Distance between Current Position and  Target Position
              if  (Distance decreases)
                - Calculate the  step size
                - Update muscle length
              end
       end
   end
end
```

Figure 5: The Algorithm that implement the Arm Inverse Kinematics. .

logic from the set of all arm actuators.

As it is possible to see the actuator length is decreased of a value D specified by equation 5 only if its decreasing allows to approach the target position, if not its antagonist actuator is considered at the next step.

$$D = -\frac{1 - e^{\frac{-(TargetP - WristP)}{K_1}}}{K_2} \quad (5)$$

As it is possible to see from equation 5 depending from the two constant K_1 and K_2 the muscle length variation decreases according to the decreasing of the distance between the wrist position ($WristP$) and the target position (TargetP).

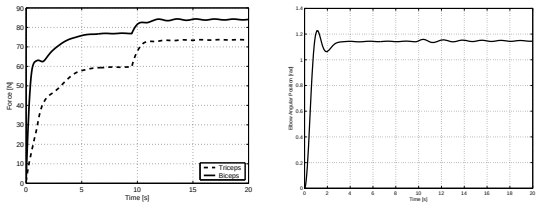
At each cycle the Direct Kinematic should be calculated and this requires a certain amount of time, in simulation this does not represent a problem, but may be critical during the normal arm operation. Nevertheless we can avoid to calculate the direct kinematics installing on the robot a vision system that can approximate the distance between the wrist and the target position. This strategy is very similar to that one performed by humans when it is required to perform a motor path in order to reach a target object. At least during the learning phase, humans being take advantage of the visual feedback to estimate the distance between the object and the hand.

8 STIFFNESS CONTROL AND TRAJECTORY FOLLOWING

Here we report the results of some simulations conducted on the robot models. The dynamic model was developed using the tool SimMechanics in Simulink environment, we chose an integration step of 1ms in order to comply with the arm dynamic. At first we tested the ability of the neural circuit in regulating the elbow stiffness, in this case a noise force was applied on the wrist. In the second part of our experiments we tested the performance of the controller in following an assigned trajectory, changing the precision we show that the algorithm time complexity will decrease exponentially.

8.1 Joint Stiffness Regulation

The control of the joint's stiffness is very important during the execution of a certain task with the robot's arm. This is true for industrial robots, but is particulary important for humanoid robots. Usually industrial manipulators operate in a protected environment where humans have a restricted access, in order to guarantee a safe operation for the robot and for the human. It is difficult to control the joint stiffness for an industrial manipulator and even if this is possible the inertia force that acts during the movement can be lethal for a human being hit by the robot. Humanoid robots are expected to operate and collaborate with humans, during a task execution, therefore the robot must not be dangerous; Humans usually can reduce or increase the joint stiffness when they are performing a certain task. For example catching a heavy object that is moving fast requires a stiffness increase of the lower and upper body articulations, while making a caress to someone requires a low stiffness for the arm's articulations. The articulation's stiffness, in turn, is regulated by the muscle cocontraction.
In the reflex modules the stiffness is regulated by the P_i signals that excite the Ins interneurons. In order to demonstrate such a capability in the reflex module I increased the P_i signals for the elbow actuators to the maximum value possible, 1. Picture 6a shows the forces increasing due to the P command.



a. b.

Figure 6: (a) Biceps and Triceps forces during the application of a dangerous force to the hand. (b) Elbow angular position during the increasing of the joint's stiffness.

The forces in the biceps and triceps actuators increase at the same time, in order to avoid the joint movement. We can observe also that the triceps increases its force more than the biceps; the important thing is that the total momentum exercised on the elbow joint is equal to zero in order to guarantee its

position (Figure 6b). As it is possible to note, the elbow position does not change when its stiffness is increased. We can also observe a collateral effect due to the stiffness increasing: for a certain period there are some little oscillations in the elbow joint. Surprisingly this phenomena is present also in humans. When the muscles are very highly co-contracted, a tremor will occur in the arm.

8.2 Trajectory Following

In the first simulation (Figure 7) the arm follows a straight trajectory from the position (0.46,-14) to the position (0.46,9), all the quantities are expressed relatively to the robot workspace. In this case we impose a maximum deviation from the reference trajectory of 6mm, the average error is 5mm. We can increase the precision, but this will also increase exponentially the computational time require to calculate the trajectory.

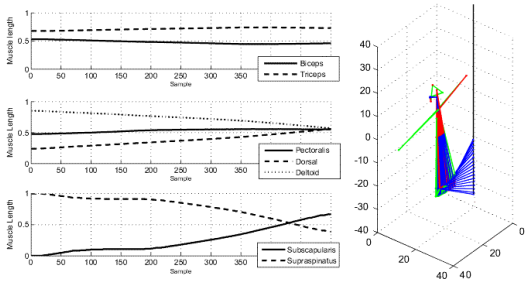


Figure 7: Following a straight trajectory.

In figure 7 are showed also the normalized values for the actuators lengths, as it is possible to note the biceps and triceps actuators lengths do not change considerably. There is instead a big upper arm rotation (relative to an axis which direction is equal to the first link direction), due to a big variation of the supraspinatus and subscapularis actuators lengths.

In the second simulation (8) we give as reference a more complex trajectory (useful for example to avoid an obstacle in the workspace), also in this case the trajectory is well followed by the wrist.

Finally figure 9 shows the trend for the computational time complexity relative to the precision in reaching the target position. As it is possible to see the trend is exponential, this means that the chose of the robot precision is a critical factor for the control system.

.

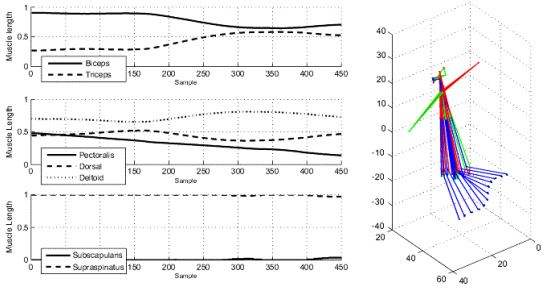


Figure 8: Following a complex trajectory.

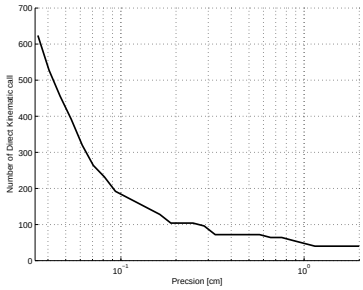


Figure 9: Trend for the time complexity.

9 CONCLUSION AND FUTURE DEVELOPMENTS

In this paper we describe the architecture and the main features of an anthropomorphic artificial arm intended for applications in the field of the Humanoid Robotics. In particular we analyzed its actuation system, that emulates the arrangement of the human musculature, and its control system that is organized in a modular and hierarchical way. The high level controller is charged to generate the arm trajectory and to perform the inverse kinematic, the low level controller sets the muscles lengths and the joints stiffness. In order to perform the inverse kinematics we used the gradient descent method, this to calculate the minimum of the distance function between the current and the target wrist position. Results show that the arm can perform different kind of trajectories allowing also to regulate, during the movement execution, the joint stiffness. This is very important especially for the execution of specific tasks in collaboration with human beings. Future works will require to test the control system on the real arm and to compare the experimental results with the simulations conducted on the arm models.

REFERENCES

Daerden, F. and Lefeber, D. (2001). The concept and design of pneumatic artificial muscles. *International Journal of Fluid Power*, 2:41–50.

Folgheraiter, M. (2003). Study of an anthropomorphic artificial arm for application in humanoid robotics. *Ph.D Thesis*.

Folgheraiter, M. and Gini, G. (2001). Human-like hierarchical reflex control for an artificial hand. *Proc. IEEE Humanoids 2001, Waseda University, Tokyo*.

Gamal, M. E., Kara, A., Kawamura, K., and Fashoro, M. (1992). Reflex control for an intelligent robotics system. *Proceeding of the 1992 IEEE/RSJ Intelligent Robots and System*, 2:1347–1354.

Hannaford, B., andChing Ping Chou, J. M. W., and Marbot, P.-H. (1995). The anthroform biorobotic arm: A system for the study of spinal circuits. *Annals of Biomedical Engineering*, 23:399–408.

Hannaford, B. and Chou, C.-P. (1997). Study of human forearm posture maintenance with a physiologically based robotic arm and spinal level neural controller. *Biological Cybernetics*, 76:285–298.

Hannaford, B., Jaax, K., and Klute, G. (2001). Bio-inspired actuation and sensing. *Robotics*, 11:267–272.

Jaax, K. N. (2001). *A Robotic muscle spindle: neuromechanics of individual and ensemble response*. Phd thesis, University of Washington.

Kawamura, K., II, R. P., Wilkes, D., Alford, W., and Rogers, T. (2000). Isac: Foundations in human-humanoid interaction. *IEEE Intelligent Systems*, 15(4):38–45.

Klute, G. K. and Hannaford, B. (2000). Accounting for elastic energy storage in mckibben artificial muscle actuators. *ASME Journal of Dynamic Systems, Measurement, and Control*,, 122(2):386–388,.

Kuntimad, G. and Ranganath, H. S. (1999). Perfect image segmentation using pulse coupled neural networks. *IEEE Transaction on Neural Network*, 10(3):591–598.

Michele Folgheraiter, G. G. (2004). Human-like reflex control for an artificial hand. *BioSystem Journal*, 76(1-3):65–74. 2004.

Omura, Y. (1999). Neuron firing operations by a new basic logic element. *IEEE Electron Device Letters*, 20(5):226–228.

Sterratt, D. C. (2001). Locust olfaction synchronous oscillations in excitatory and inhibitory groups of spiking neurons. *Emergent Neural Computational Architectures*, 2036:270–284.

Toshiki Niino, Saku Egawa, H. K. and Higuchi, T. (1994). Electrostatic artificial muscle: Compact, high-power linear actuators with multipli-layer structures. *Micro Electro Mechanical Systems, 1994, MEMS '94, Proceedings, IEEE Workshop on*, pages 130–135.

DISTRIBUTED CONTROL ARCHITECTURE FOR AUTOMATED NANOHANDLING

Christian Stolle
Division of Microrobotics and Control Engineering, University of Oldenburg, 26111 Oldenburg, Germany
stolle@informatik.uni-oldenburg.de

Keywords: Automation, nanohandling, robot control.

Abstract: New distributed control architecture for micro- and nanohandling cells is presented. As a modular system it is designed to handle micro- and nanorobotic automation tasks at semi- up to full automation level. The architecture includes different visual sensors as there are scanning electron microscopes (SEM) and CCD cameras for position tracking as well as non-optical force, temperature, etc. sensors for environmental control. It allows usage of multiple nanorobots in parallel for combined autonomous fabrication tasks. The system provides a unified framework for mobile platforms and linear actors.

1 INTRODUCTION

Handling of micro- and nanoscaled objects is an important research field in micro system technology (MST) and nanotechnology. While most MST techniques are concerned with bottom-up batch procedures for massive parallel production of micro systems (Menz, et al., 2001) micro- and nanorobotics tackles the nanoassebly task top-down by applying adapted macro scale manufacturing and control methods (Fatikow and Rembold, 1997).

This paper introduces general purpose control architecture for automated robot-based handling.

1.1 Microrobot Automation

Automation in microrobotics encounters many of the problems, which have been well studied in the domains of industrial robotics and autonomous service robotics for decades. Some of the problems are e.g. collision avoidance in path planning, error handling due to uncertainty of operations (Bennewitz and Burgard, 2000), or timing constraints which need to be met for successful automation.

However, there are some environmental challenges while operating in the vacuum chamber of an SEM. It takes several minutes to generate a high vacuum (10^{-3} - 10^{-7} hPa) which is a serious time constraint. Therefore, all cell parts of a nanofabrication unit need to be inside of the SEM before operation starts, including tools and objects to be handled.

All tools, sensors and objects have to be vacuum compliant. Depending on the kind of manipulation, the materials need to be selected such that no contamination of the workpieces can occur. E.g. actors need to have low or even no abrasion.

For real-time position tracking of nanoobjects like carbon nanotubes (CNTs, typical diameter about 0.4 – 100 nm) only SEM images are available due to their high scanning rate and resolution. High scanning rates are crucial for image acquisition in closed-loop control, but they can only be archived with the tradeoff of noisy image data. All this makes real-time SEM tracking a challenging two dimensional problem (Sievers and Fatikow, 2005).

Another challenge is that there is only limited depth information available due to SEM by now. As a result approaching and depth alignment of objects and tools need to done in a try an error manner.

Automation chain planning also needs to take the different environment and object scale into account. In contrast to large scale objects it is harder to release an object than to grip it. The reason for this "sticky finger" effect (Fearing 1995) is that adhesive forces are stronger at the scale of the gripper jaws and samples than gravity. Therefore pick and place operations need to be carefully planned.

One possible solution to overcome this problem is to use electron beam induced deposition (EBiD). One end of the gripped CNT gets fixated to the specimen holder by the deposited material of the

EBiD process just before releasing it from the gripper (Wich, *et al.,* 2006).

2 CONTROL OF MICROROBOTS

Several microrobots have been developed for coarse and fine positioning to carry tools and objects. The major categories are mobile platforms and fixed platforms. Mobile robots are more flexible than fixed ones due to their working range, but they are harder to control (Ritter, *et al.*, 1992; Hülsen and Fatikow, 2005). Linear actors are most often combined to a Cartesian microrobot for (x,y,z)-coarse positioning and additional fine positioning axes.

The control system of the nanohandling cell in the SEM is split into two parts: high-level control and low-level control. Low-level control continuously compares set-point and actual value and calculates parameters for actor steering signals. Low-level controllers are therefore responsible for the control of states and can process tasks as “grip object” or “move actor from point A to point B”.

Two types of controllers are used: open- and closed-loop controllers. Closed-loop controllers have continuous feedback of the current system state through sensor data while open-loop controllers just execute a command due to their internal knowledge of the current state.

High-level control is responsible for planning and execution of automation tasks and for supporting user input for tele-operation. Error handling of low-level tasks, parallelization of automation tasks and path planning are also part of high-level control.

The user interacts with the control system via a graphical user interface (GUI), which is displaying the current system state and is forwarding user input to high-level control.

Several different sensors measure continuously data within the vacuum chamber (e.g. cameras, pressure sensors, etc.). These measurements are collected and provided to low- and high-level control. Low-level control directly derives set values (e.g. actuation signals) from the sensor data, which are applied to the actuators. The update rate of the sensor data has direct influence on positioning speed and accuracy of the robot. High-level control sets configuration values to sensors and actor signal generators according to the task that needs to be performed. It also provides input for low-level control as position data values which are used as goal positions for low-level closed-loop control (Fig. 1).

The reliability of closed-loop low-level control nowadays has increased up to a level where single tasks can be done semi-automatically, and even full automation of various handling steps can be taken into account. The latter is a hierarchical process which takes place at the high-level control layer, where the sensors for tracking can be selected, and the sequence of process primitives is controlled. Execution and reliability of single process primitives is the responsibility of the low-level control layer (Fatikow, *et al.*, 2006).

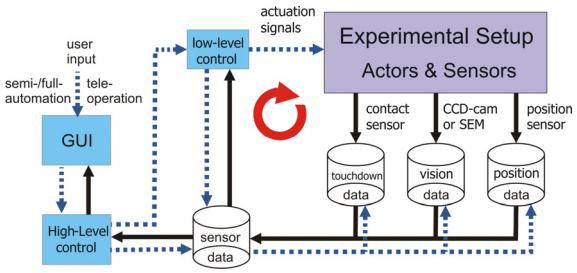


Figure 1: Control dataflow with different sensors for continuous feedback of the current system state. Actuation signals are generated by low-level control from sensor input, high-level control processes user input and sends steering signals to all components.

3 CONTROL ARCHITECTURE

3.1 Control Architecture Requirements

The rough dataflow schema for experimental setups in figure 1 needs to be met by any control architecture. All the actuators and sensors of the cell need to be included into a modular control system for successful automation. In order to be independent of a particular experimental setup this control system architecture needs to fulfill certain requirements:

- New actors can be integrated with low effort, independent of the low-level control algorithm of the actor.
- New sensors can be integrated with low effort, without the need of restructuring other parts of the architecture.
- Sensors and actors can be accessed through a common interface to prevent changes in high-level control.

- The time from data acquisition till actuation needs to be sufficiently low to be able to archive the required low-level control results.
- Communication between units needs to be done asynchronous in order not to block parts of the systems while waiting for others.
- All commands issued by high-level control have to be executed providing feedback about the operation result for reliable error handling.
- The architecture is able to be scaled up, such that enough actors and sensors can be integrated for any automation sequence.
- The architecture has to allow parallel execution of several automation tasks.
- Depending on the process' power requirements parts of the architecture may be distributed throughout several PCs.

This set of requirements leads to a distributed system on a common client server basis, described below.

3.2 Distributed Control Architecture

The system architecture applied consists of high-level control, sensor, vision and several low-level control server (Fig. 2) following a "Black Box Design" (Fatikow, *et al.* 2006). Every server offers an individual service which is defined by a public interface. Therefore every component in this modular control system can be easily replaced or updated independently of the other components.

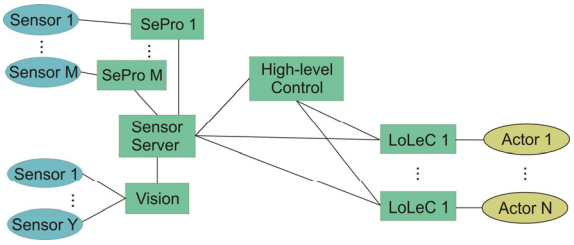


Figure 2: Software architecture connection chart. Rectangles are servers (e.g. sensor programs SePro, low-level controller LoLeC, etc.) and circles are hardware components. Automation module is part of the high-level control server.

Visual feedback is provided by the vision server. It is responsible for collecting images from different sources and extracts position and orientation data (poses) of tracked objects in real-time. These poses are transmitted to the sensor server. All sensor data (e.g. pose, touchdown, force, temperature, etc.) are collected by the sensor server which is supplied by different sensor service programs (e.g. vision server). This data is provided to low- or high-level control, which act as client of the sensor server. It represents therefore an abstraction layer for sensors and builds a common interface for further processing. The different sensors may acquire their data at different update rates.

In contrast to the previous architecture (Fatikow, *et al.* 2006) the newly developed one has one low-level control server for every single actor. Each of the low-level control server requests the data needed for closed-loop control from the sensor server. There is a uniform interface for commands from high-level control. The advantage of this lean low-level control server approach is that single servers can be easily distributed among several PCs.

Their good maintainability is another important feature. Furthermore, it is easy to include different – e.g. fixed and mobile – robot platforms, because only the internal structure of the low-level controller has to be changed.

On the high-level control level still only poses have to be submitted through the command interface. Actors with internal sensors are included into this architecture in a way that the low-level control server provides the internal sensor readings to the sensor server and receives high-level commands.

In order not to adapt the high-level control server to every change in an automation sequence, a script language has been developed, which is described in more detail in section 4.2. These scripts get interpreted by the high-level control server and then mapped to low-level control tasks and steering signals (Fig. 1). Low-level control tasks can either be closed-loop for positioning or open-loop (e.g. gripping objects) tasks. The latter have to be monitored to check the operation results.

The proposed system architecture uses the Common Object Request Broker Architecture (CORBA) as communication framework, which is an object oriented middleware defined by the Object Management Group (OMG), and which is platform- and language-independent. There are several implementations of the CORBA standard, including some for real-time applications.

CORBA's Interface Definition Language (IDL) is used as common language for all communication interfaces between different network components. Therefore the servers and clients in the system architecture only need to implement the required IDL interfaces to be able to communicate to each other via simple method calls.

A major advantage of IDLs is that they can be translated into different programming languages,

which enables heterogeneous software design. The communication overhead in a closed loop cycle has been evaluated to be sufficient low (< 5 µs) in a local full switched Ethernet network. Therefore the limiting time factor still remains to be the image acquisition time.

The interfaces of the client server architecture are designed to provide asynchronous communication. All control commands return unique process IDs so that delayed control feedback can be matched with the corresponding command. This part of the control feedback is crucial for successful, reliable automation. The automation scheduler will wait at synchronization points for this feedback (Section 4.1). To avoid dead-locks a time-out is issued after a sufficient time. The system has also a common time base enabling low-level control servers to decide whether sensor data is outdated or not. The clock synchronization is periodically triggered by a master clock.

Every network component is designed in a way that it can be run on different PCs, which makes the system fairly flexible. The distribution of low-level control servers is especially useful for control cycles that run in parallel because the distributed controllers do not compete for the same PC hardware resources. The control architecture may contain several sensor servers for different data, to overcome possible data acquisition bottle necks. It is possible to organize the sensor data traffic shaping a minimum update interval so that the inbound traffic of any single sensor server can be controlled.

The problem of keeping several low-level control programs maintainable is tackled by common low-level server templates which are available for different actor types. This also enables rapid design and integration of new actors. Similar templates will also be provided to sensor data acquisition server.

4 HIGH-LEVEL CONTROL

The purpose of high-level control is to processes an automation sequence or to receive tele-operation commands from a user via graphical user interface (GUI). Input information is translated into low-level command tokens called tasks and steering signals for the selection of certain sensors.

4.1 High-Level Control Design

Based on the description above the high-level controller provides tele-operation, automation, path planning, error handling and parallel execution of tasks. These different tasks are addressed in different units in our high-level control design (Fig. 3).

Main trigger for high-level control is a human controller, who inputs a tele-operation, semi- or full-automation command using the GUI.

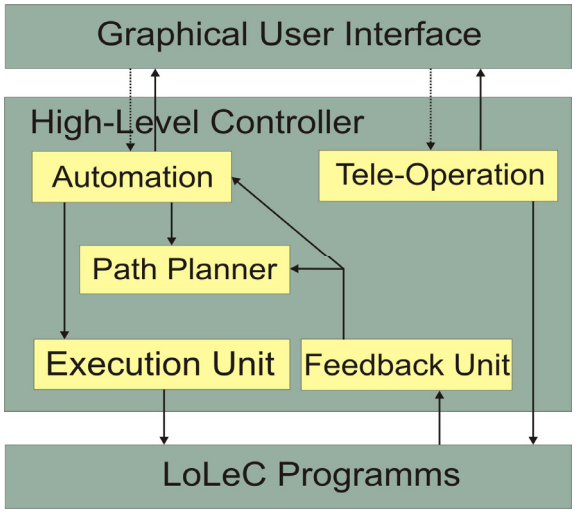


Figure 3: Components of a high-level controller.

If an automation command is received by the automation units of the high-level controller e.g. start or stop automation, a preloaded automation sequence is processed. The sequence of automation tasks is executed one by one or in parallel.

Every task has a defined set of pre- and post-conditions (Section 4.2). The automation unit decides based on required resources (e.g. sensors and actors) and pre- and postconditions, whether two consecutive tasks can run in parallel. These postconditions should not be contradictive. If resource conflicts arise, a barrier approach is taken so that all parallel tasks are finished before automation of the resource critical task starts (Fig. 4).

For SEM automation these resource-critical tasks are often tasks which use the SEM as tracking sensor for two objects on different heights or if the SEM acts as sensor and actor at the same time (e.g. EBiD and a concurrent positioning task).

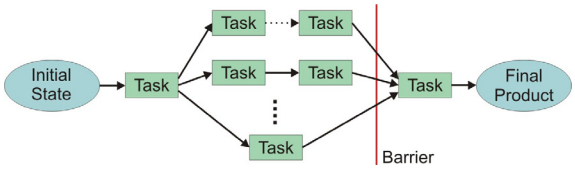


Figure 4: Schematic view on an arbitrary automation task. Parallel tasks are finished before resource critical tasks get started.

At the automation unit level only goal positions are visible. The main task of the path planning unit is to break down goal positions into a discrete set of intermediate positions that approximate the trajectory of the robots. This trajectory should be time efficient since bad trajectories may lead to poor automation cycle times (e.g. rotating to the left might be slower that moving in x, y direction.). Another goal of path planning is to avoid collisions with objects or other robots. For collision avoidance an environmental model is required. Path planning can be performed during the automation sequence design phase or online.

On the current development state of the system only offline path planning is supported, which is sufficient in most cases but delegates more responsibility to the designer of the automation sequence.

Subgoal positions calculated by the planning unit are sent to the execution unit as well as steering commands of the automation unit. The execution unit is responsible for execution of single actuation commands.

First for every command received, all necessary steering signals are sent to the involved programs. These steering signals prepare all components in the system for the low-level control command that follows. For example the right camera is selected or actor subcomponents are turned on or off.

After receiving an acknowledgment for every the steering signal the low-level control command is issued to the corresponding LoLeC.

One of the major problems in micro- and nanorobotic is the high error rate of single automation tasks. Due to environmental and scale effects operations that are simple in macro automation as positioning do have a high error rate. E.g. there might well be endless positioning tries because of a too low low-level control error threshold, which was correct at a different humidity or temperature level.

$$\varepsilon_{seq} = 1 - \prod_{t \in tasks} (1 - \varepsilon_t) \qquad (1)$$

As in macro automation the error of a sequence is the multiplied error rate of every single automation task (Eq. 1). E.g. for a sequence consisting of 7 task each having an complimentary error rate of 10% the error rate of the automation sequence is higher then 50%.

The first step to deal with these high error rates is a reliable error detection which is the task of the feedback unit in the high-level controller. This unit receives all error conditions which occurred in the components of the system (e.g. a LoLeC) and presents them to the right abstraction level.

Another yet not implemented task for the feedback unit would be the observation of certain environmental states. E.g. a carbon nanotube (CNT) escaping from a gripper needs to be detected through visual feedback since there is no other sensorial information for this event. To cover this kind of errors the feedback unit could track the CNT through a vision sensor and rise an error condition if the CNT gets lost.

Error conditions get handled by different units in a different manner. If a collision is detected the planning unit might generate a new trajectory.

On automation unit level the error handling is performed by sweeping backwards in the automation sequence and looking for the first task with pre-conditions that meet the current system status.

If there is no possible way of handling an error automatically the automation sequence is interrupted and needs to be handled by a human controller.

The lowest level of automation i.e. tele-operation is provided by the tele-operation unit, which directly receives commands from a human controller. These commands are directly provided to the corresponding LoLeC servers. It is in the responsibility of the human controller to avoid collisions and move the robots to certain positions using “save” trajectories.

4.2 Script Language for Automation Sequences

Modeling a suitable automation sequence for a given handling task can be performed in three steps. First, the hardware setup has to be defined according to the requirement analysis. Then robot-based process primitives and their pre- and post-conditions have to be defined (e.g. move robot to target position if it is in range of the vision sensor). Finally, an automation sequence is to be found, which meets all pre- and post-conditions of the process primitives, avoids collisions and eventually accomplishes the automation task. Additional constraints as e.g. executing time can be taken into account as well.

The differences in process primitives between different nanohandling robot cells (hardware setups) impose the problems of how to avoid reimplementation and hard coding of process primitives and how to change the automation sequence without changing the program code.

A flexible script language has been developed, following the script based approach of Thompson and Fearing (2001). The different commands of the

language are the process primitives itself. They are implemented as sub-classes of a common task base class. This avoids reimplementation of common concepts as error handling or message protocols.

Composition and iteration are provided as general language constructs. Positions as e.g. parking or working positions for different actors can be defined as constants and later on be used in all commands.

Based on this script language arbitrary automation sequences on the predefined operators can be defined as plain text files (Fig. 5).

These sequence files gets interpreted and executed by the high-level controller automation unit. This way the automation sequences can take advantage of the underlying closed-loop control. The concept enables rapid development of different automation sequences for the same set of process primitives, while the high-level control program keeps unchanged.

```
lift(TOUCHDOWN_DIST);
move(EC,WORKING_POS);
grip(TRUE);
lift(SECURITY_DIST);
move(DROP_POS);
```

Figure 5: Automation sequence that lifts the specimen holder, moves the robot into the focus of the electron beam an grips an object.

The language design has been chosen with regard to future application of a planning algorithm like Metric-FF (Hoffmann, 2003) to find the optimal automation sequence for given automation task.

5 CONCLUSIONS AND FUTUREWORK

This paper has presented a new distributed system architecture for controlling micro- and nanorobotic cells competitive to the system of Fatikow et al. (2006). However it scales up easier due to the lean low-level control design.

In contrast to the micro handing setup in Thompson and Fearing (2005) this system architecture is designed for micro- and nanohandling and full automation inside and outside a SEM chamber.

Aside of all positive design aspects the full capabilities of the system needs to be evaluated carefully and applied to different kind of nano-handling cells. So far only partial tests of components and component interaction have been performed.

Nevertheless subcomponents as there are the high-level controller automation unit, the sensorial parts and two low-level controllers for linear actors have shown decent control behavior.

More attention also needs to be paid to automation tasks reliability and detection of error conditions apart from positioning tasks as described in 4.1.

REFERENCES

Bennewitz, M., Burgard, W., 2000. A Probabilistic Method for Planning Collision-free Trajectories of Multiple Mobile Robots. *ECAI'00. Proc. of the workshop Service Robotics - Applications and Safety Issues in an Emerging Market at the 14th European Conference on Artificial Intelligence*

Fatikow, S., Rembold, U., 1997. *Microsystem Technology and Microrobotics,* Springer-Verlag

Fatikow, S., Wich, T., Hülsen, H., Sievers, T., and Jähnisch, M. 2006. Microrobot system for automatic nanohandling inside a scanning electron microscope. *ICRA'06. Proc. of Int. Conference on Robotics and Automation*, Orlando, FL, USA

Fearing, R.S., 1995. Survey of Sticking Effects for Micro-Parts, *IROS'95. Proc. of IEEE Int. Conf. on Robotics and Intelligent Systems, Pittsburg, USA*

Hoffmann, J., 2003. The Metric-FF Planning System: Translating ``Ignoring Delete Lists" to Numeric State Variables. *Journal of Artificial Intelligence Research Vol. 20, pp. 291-341*

Hülsen, H., Fatikow, S., 2005. Extrapolation with a self-organising locally interpolating map. *ICINCO'05. Proc. of Int. Conference on Informatics in Control, Automation and Robotics, Barcelona, Spain, pp. 173-178*

Menz, W., Mohr J., Paul O., 2001. *Microsystem Technology*, Wiley-VCH.

Ritter, H., Martinetz, T., Schulten, K., 1992. Neural computation and selforganizing maps. *Addison Wesley, Reading, Mass.*

Sievers, T., Fatikow, S., 2005. Visual Servoing of a Mobile Microrobot inside a Scanning Electron Microscope, *IROS'05. Proc. of IEEE Int. Conference of Intelligent Robots and Systems, pp. 1682-1686*

Thompson, J. A., Fearing, R. S., 2001. Automating microassembly with ortho-tweezers and force sensing. *IROS'01. Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems), Maui, HI, pp. 1327-1334*

Wich, T., Sievers, T., and Fatikow, S., 2006, Assembly inside a Scanning Electron Microscope using Electron Beam induced Deposition, *IROS'06. Proc. of IEEE Int. Conference on Intelligent Robots and Systems, pp. 294-299*

FEASIBILITY OF SUBSPACE IDENTIFICATION FOR BIPEDS
An Innovative Approach for Kino-Dynamic Systems

Muhammad Saad Saleem and Ibrahim A. Sultan
School of Science and Engineering, University of Ballarat, Mount Helen, Victoria, Australia
msaleem@students.ballarat.edu.au, i.sultan@ballarat.edu.au

Keywords: Biped, subspace identification, kino-dynamic, operational space control, biped stability, crisp control.

Abstract: Different approaches have been briefly overviewed which have been used in stability of biped robots. Current implementations either mimic human behavior or use heuristic control. This paper suggests the use of model-free crisp control in operational space configuration for better control and understanding of kino-dynamic systems and biped robots.

1 INTRODUCTION

Designing a control strategy for a biped robot can be quite tedious as dynamics involved are non-linear, multi-variable, naturally unstable and foot-ground interaction is limited (Wolkotte (2003); Kim et al. (2004); Caballero et al. (2004)). These all problems suggest that controller should be sophisticated enough to cater for all these factors. This is why most implementations don't use classical control techniques but rely on techniques which mimic human behavior or are based on heuristic control (Pratt (2000)).

In order to examine crisp control and a mathematical solution for biped stability using other than above mentioned techniques, subspace identification is proposed, which then can be coupled with post-modern control techniques such as $\mathscr{H}_\infty$ to design a model-free control system. In theory such controllers are developed but has never been used for kino-dynamic systems (Favoreel et al. (1998); Woodley et al. (2001a)).

In the paper, previous implementations of biped robots are mentioned in section 2. Model-free and model based implementations are briefly discussed in section 3. Subspace identification and its model-free implementations are discussed in section 4. Proposed implemetation is mentioned in section 5. Section 6 discusses the results of using subspace identification technique in biped robot leg.

2 PREVIOUS IMPLEMENTATIONS

An overview of literature suggests that history of biped robots has only handful of milestones. Quasi-dynamic walking gait on bipeds was achieved in 1980 by Kato et al. using artificial muscles (Kato et al. (1983)). In 1983, Raibert demonstrated a planar one-legged hopping robot that could hop at desired velocity and jump over small obstacles (Raibert (1986)). In 1990, McGeer demonstrated first passive walking for robots that could walk down a slop without any active elements (McGeer (1990)). In 1997, Honda introduced its biped robot P2 which set a new trend in bipeds. Latest from Honda, ASIMO, has state-of-the-art technology in this field (Sakagami et al. (2002); Hirai et al. (1998); Lim and Takanishi (2005)). Control systems employed in the development of bipeds can be divided into different categories.

Most of the robots fall into the category which employs simple models, which can be calculated by Newtonian mechanics, others are based on walking and running dynamics (Kajita et al. (1992); Schwind (1998)). These models are because of the inspiration from biometrics (McMahon (1984); Alexander (1996)). This technique is best used when trajectory is given. It can be subdivided into further two types. First one are the bipeds which are clone of ASIMO and others are based on intuitive control. The best ef-

fort in this technique has come from Pratt and Pratt, and most impressive implementations in this type of control framework also came from the same group (Pratt and Pratt (1998, 1999); Pratt (2000)).

Other type of controllers are based on "neural" oscillators or pattern generators (Taga (1995)). There are studies which suggest that vertebrates have some kind of pattern generation mechanism which enables them to walk dynamically. Generators can be hand-tuned to construct a detailed feedback response for dynamic walking. Last type of controllers are the ones which are based on machine learning.

3 METHODS TO USE EXPERIMENTAL DATA

To design a control system, equations for a biped robot can be calculated from Newtonian mechanics. It is shown in (Pratt (2000)) that equation of motion of a massless leg with a torso having mass m, can be written as:

$$ml^2\ddot{\theta}_1 = mgl\sin\theta_1 - 2ml\dot{l}\dot{\theta}_1 - J\ddot{\theta}_b \qquad (1)$$

here θ_1 is the angle between the normal axis to ground and axis going through the CoP (center of pressure) in foot and center of mass of the torso, J is the rotational moment of inertia, and θ_b is the angle between torso axis and the leg. Equation 1 suggests that there are three ways to change rotational dynamics about center of pressure. First method is to change the position of the body, which will change θ_1 and location for center of pressure. This method is the most effective one. Second method is to change the inertial momentum J and third method is to change the length l. Effect because of the last two quantities is not much when compared with effect due to change in location of center of pressure.

As we are more interested in exploring a more robust and generic solution for kino-dynamic systems, techniques to use experimental data to determine system equations will be discussed. There are four methods to use experimental data as shown in table 1 (Woodley (2001)). Mainly, choice depends on application. For real-time systems which are easy to model, indirect control is a better choice. The system then adapts itself and updates its model parameters according to the conditions. Normally on-line model based design is referred as indirect control. If a system is hard to model from first principles (as Newton's laws of motion) or there are time varying nonlinearities then direct adaptive control would suite the application. Examples of plants which are difficult to model are arc furnaces (Wilson (1997); Staib and Staib (1992)) and helicopter rotors (Lohar (2000); Tischler et al. (1994)). Biped robots on the other hand can be modeled but they exhibit time varying nonlinearities (Wolkotte (2003); Kim et al. (2004); Caballero et al. (2004)).

4 SYSTEM IDENTIFICATION

There are many system identification techniques. The list starts with classical prediction error (PE) and its variants; auto regression with exogenous input (ARX), output error (OE), auto regression moving average with exogenous input (ARMAX), and Box Jenkins (BJ) (Norton (1986); Ljung (1999)).

4.1 Subspace Identification

Aside from classic system identification methods, there are subspace identification methods, which gained a lot of popularity in recent years (Morari and Lee (1999)).

If plant's input and output values at discrete times are given by (Overschee and Moor (1996)):

$$\left(\begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{n-1} \end{bmatrix}, \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{n-1} \end{bmatrix} \right)$$

Hankel matrices for past and future inputs are written as

$$U_p \triangleq \begin{bmatrix} u_0 & u_1 & \cdots & u_{j-1} \\ u_1 & u_2 & \cdots & u_j \\ \vdots & \vdots & \cdots & \vdots \\ u_{i-1} & u_i & \cdots & u_{i+j-2} \end{bmatrix} \in \mathbb{R}^{im\times j}$$

$$U_f \triangleq \begin{bmatrix} u_i & u_{i+1} & \cdots & u_{i+j-1} \\ u_{i+1} & u_{i+2} & \cdots & u_{i+j} \\ \vdots & \vdots & \cdots & \vdots \\ u_{2i-1} & u_{2i} & \cdots & u_{2i+j-2} \end{bmatrix} \in \mathbb{R}^{im\times j}$$

Similarly Hankel matrices for past and future outputs can be written as $Y_p \in \mathbb{R}^{il\times j}$ and $Y_f \in \mathbb{R}^{il\times j}$ respectively. Let us define W_p as

$$W_p \triangleq \begin{bmatrix} U_p \\ Y_p \end{bmatrix}$$

Linear least squares predictor of Y_f with given W_p and U_f can be written as Frobenius norm minimization

$$\min_{L_w, L_u} \left\| Y_f - \begin{bmatrix} L_w & L_u \end{bmatrix} \begin{bmatrix} W_p \\ U_f \end{bmatrix} \right\|_F^2$$

Table 1: Four different techniques of control design from experimental data.

	With Plant Model	Without Plant Model
Online	Indirect Adaptive	Direct Adaptive
Offline	Model Based Design	Direct Control Design

From subspace orthogonal project, L_w and L_u is calculated as

$$\begin{bmatrix} L_w & L_u \end{bmatrix} = Y_f \begin{bmatrix} W_p \\ U_f \end{bmatrix}^{\mathrm{T}} \left(\begin{bmatrix} W_p \\ U_f \end{bmatrix} \begin{bmatrix} W_p \\ U_f \end{bmatrix}^{\mathrm{T}} \right)^{\dagger} \quad (2)$$

where † denotes pseudo-inverse. Future outputs can be predicted from past inputs, outputs, and future inputs.

$$\begin{bmatrix} \hat{y}_k \\ \vdots \\ \hat{y}_{k+i-1} \end{bmatrix} = L_w \begin{bmatrix} u_{k-i} \\ \vdots \\ u_{k-1} \\ y_{k-i} \\ \vdots \\ y_{k-1} \end{bmatrix} + L_u \begin{bmatrix} u_k \\ \vdots \\ u_{k+i-1} \end{bmatrix} \quad (3)$$

Pseudo-inverse is normally calculated through Singular Value Decomposition (SVD) but Woodley et al. presented another way by using Cholesky factorization, which is computationally faster and consumes less memory (Woodley et al. (2001b)). It has already been used for guidance and control of unmanned vehicle (Kelbley (2006)).

4.2 Advantages of Subspace Identification Methods

Subspace Identification Methods (SIM) have many advantages over classical system identification techniques (Overschee and Moor (1996)). Notables are:

- From plant's input and output data, a predictor is found same as Kalman filter states, which makes it a simple least square problem. The whole architecture is streamlined and user-friendly.
- When implemented in direct adaptive control, plant model is not needed to be simplified, which can omit useful information from plant, as in SIM, all the plant information is stored in a compact form of subspace predictor.
- Output of subspace identification methods is in state space form which makes it easy to implement in computer but it's architecture has been exploited in different model free implementations as well (Woodley et al. (2001b); Favoreel et al. (1999b,a)).

Wernholt used SIM to solved system identification problem for ABB IRB 6600 robot (Wernholt (2004)). Hsu et al. used N4SID in style translation for human motion. These are some of the examples that how SIMs are being used.

4.3 Reported Problems in Subspace Identification Methods

There are a few problems in subspace identification methods. Many of these problems have been discussed in recent literature and partial remedies have been suggested (Chou and Verhaegen (1997); Lin et al. (2004); Wang and Qin (2004); Chiuso and Picci (2005)). Some of these problems are:

- Biased estimate for closed loop data.
- Errors-in-variables situation due to a projection performed in the algorithm.
- Assumption of noise-free input.

It is expected that in direct adaptive system, which calculates plant's model and designs controller in realtime, this problem will not faced but final answer to this can only be given after its implementation.

4.4 Types of Subspace Identification Methods

There are many implementations of subspace identification methods. Notables are:

- Canonical variate analysis (CVA) (Larimore (1990)).
- Multivariable output-error state space (MOESP) (Verhaegen and Dewilde (1992)).
- Numerical algorithms for subspace state space system identification (N4SID) (Overschee and Moor (1994)).
- Eigensystem realization analysis (ERA) (Juang (1994)).
- Subspace fitting (Jansson and Wahlberg (1996)).
- Stochastic subspace identification method using principal component analysis (SIMPCA) (Wang and Qin (2004)).

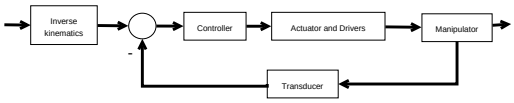


Figure 1: Joint space control.

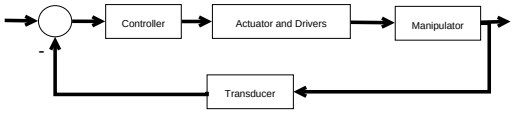


Figure 2: Operational space control.

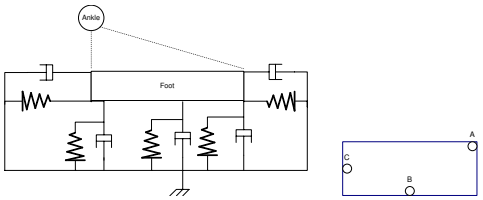


Figure 3: Foot ground interaction. On the left is the side view and on the right is the top view of foot model where points A, B, and C are connected to three dampers and springs. Dampers and springs connected on sides are responsible for friction with the ground.

5 PROPOSED IMPLEMENTATION

Joint space control is consisted of two subproblems. First, manipulator inverse kinematics is performed and then joint space control scheme is devised which allows the end effector to follow the reference input. The main computational burden in this scheme is because of inverse kinematics, which is normally performed by using different optimization techniques, as in a redundant system, there can be infinite solutions for a given task (Lope et al. (2003); Gupta et al. (1993); Kim et al. (2003)). Many implementations of joint space control can be found in the literature (Laib (2000); Kelly (1997); Arimoto (1995); Kelly (1993); Wen et al. (1992); Tomei (1991); Takegaki and Arimoto (1981); Zhang et al. (2000)).

In many applications, desired path of end effector is specified in operational space. Operational space control, on the other hand, is used for constrained manipulator motions (Sciavicco and Siciliano (2000); Sapio and Khatib (2005)). These constraints can be because of gravity or kinematically imposed. It can be seen in figure 2 that inverse kinematics is embedded in the closed-loop control law but not explicitly performed as shown in figure 1 (Sciavicco and Siciliano (2000)). Operational space control and task space control sometimes allude to the same concept (Khalil and Dombre (2004); Xie (2003); Sciavicco and Siciliano (2000)). Sapio and Khatib has simulated operational control schemes in physiological model of a human body under constrained conditions (Sapio and Khatib (2005)).

6 EXPERIMENT

MATLAB®and Simulink®by MathWorks Inc. have been employed to simulate a bipedal leg with torso. Under the action of normal gravity and exogenous force signals at each joint, the leg falls down and trajectory of torso is recorded. Using Subspace Identification, a predictor is found. This predictor is then applied on input joint signals. First, predicted trajectories are presented and then trajectories are predicted by updating previous trajectory from actual outputs after every prediction.

Following algorithm gives error between actual and predicted trajectories:

1. Prediction horizon, *i* is chosen and experiment is performed with given input and resultant torso trajectory is noted
2. From noted trajectory, a predictor is calculated using subspace projection algorithm
3. Outputs are calculated using subspace predictor and given inputs at joints
4. Difference between calculated values and actual values are plotted for each axis
5. Prediction horizon is changed and the whole process is repeated

One of the challenges in simulations was to simulate foot-ground interaction. Many implementations can be found in the literature (Hsu et al. (2005); Ogihara and Yamazaki (2001); Wang (2005); Wolkotte (2003)). Model with three contact points was devised after inspiration from human foot. This is shown in figure 3.

6.1 Assumptions

It is assumed that there are only three points where foot can touch the ground as shown in figure 3 and there is no air friction.

6.2 Results

For prediction horizon *i* less than a certain value, the system simply fails to predict the future outputs. Some suggest that the value of *i* should be 2 to 3 times

Table 2: Supposed values of different parameters for simulation.

	Length or radius [m]	Width [m]	Height [m]	Mass [kg]
Torso	0.1	0.4	0.5	20
Thigh	0.05		0.4	10
Calf	0.05		0.4	5
Foot	0.3	0.07	0.3	2
	Shape	I_1 [kg m^2]	I_2 [kg m^2]	I_3 [kg m^2]
Torso	Parallelepiped	0.688	0.4333	0.2833
Thigh	Cylinder	0.2333	0.2333	0.05
Calf	Cylinder	0.0698	0.0698	0.0063
Foot	Parallelepiped	9.6667e-4	0.0151	0.0158

the expected order of the system for stable and accurate results (Woodley (2001)), however, there is no hard and fast rule. In our experiments, the prediction horizon more than 10 did not improve the accuracy of the prediction. Increasing the value of i can also be computationally expensive as even with Cholesky/SVD factorization technique, the complexity of finding a subspace predictor is $O(ij+i^3)$, where j is number of prediction problems (Golub and Loan (1996)). It can be seen in the simulation and graphs that for movements of more than 1 meter, the error is in the order of micrometers. These results are very encouraging especially when there are multiple rigid bodies which are coupled together with rotatory joints and ground-foot interaction is present with given friction.

7 FUTURE WORK

To find subspace predictor, Hanekel matrix structure can be exploited for a better real-time operation. This work can be extended to a complete implementation of a model-free control system such as the one suggested by Woodley et al.. One of the challenges in the actual implementation is determination of uncertainty block Δ for the given system using techniques such as model unfalsification but without excessive overload of high computations (Woodley et al. (1998), Paul B. Brugarolas (2004)).

REFERENCES

Alexander, R. M. (1996). *Optima for Animals*. Princeton University Press, revised edition.

Arimoto, S. (1995). Fundamental problems of robot control: Parti, innovations in the realm of robot servo-loops. *Robotica*, 13:19–27.

Caballero, R., Armada, M. A., and Akinfiev, T. (2004). Robust cascade controller for nonlinearly actuated biped robots: experimental evaluation. *International Journal of Robotics Research*, 23(10/11):1075–1095.

Chiuso, A. and Picci, G. (2005). Consistency analysis of some closed-loop subspace identification methods. *Automatica*, 41(3):377–391.

Chou, C. T. and Verhaegen, M. (1997). Subspace algorithms for the identification of multivariable dynamic errors-in-variables models. *Automatica*, 33:1857–1869.

Favoreel, W., Moor, B. D., Gevers, M., and Overschee, P. V. (1998). Model-free subspace-based LQG-design. Technical report, Katholieke Universiteit Leuven.

Favoreel, W., Moor, B. D., Gevers, M., and Overschee, P. V. (1999a). Closed loop model-free subspace-based LQG-design. In *Proceedings of the IEEE Mediterranean Conference on Control and Automation*, Haifa, Israel.

Favoreel, W., Moor, B. D., and Overschee, P. V. (1999b). Model-free subspace-based LQG-design. In *Proceedings of the American Control Conference*, pages 3372–3376.

Golub, G. H. and Loan, C. F. V. (1996). *Matrix Computations*. The Johns Hopkins University Press.

Gupta, M. M., Rao, D. H., and Nikiforuk, P. N. (1993). Dynamic neural network based inverse kinematics transformation of two- and three-linked robots. In *12th World Congress, International Federation of Automatic Control, Sydney, Australia*, pages 289–296.

Hirai, K., Hirose, M., Haikawa, Y., and Takenaka, T. (1998). The development of honda humanoid robot. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1321–1326.

Hsu, E., Pulli, K., and Popović, J. (2005). Style translation for human motion. *ACM Transactions on Graphics (TOG)*, 24(3):1082–1089.

Jansson, M. and Wahlberg, B. (1996). A linear regression approach to state-space subspace system identification. *Signal Processing*, 52:103–129.

Juang, J. N. (1994). *Applied System Identification*. PTR Prentice-Hall.

Kajita, S., Yamaura, T., and Kobayashi, A. (1992). Dynamic walking control of a biped robot along a potential energy conserving orbit. *IEEE Transactions on Robotics and Automation*, 6(1):431–438.

Kato, T., Takanishi, A., Jishikawa, H., and Kato, I. (1983). The realization of the quasi-dynamic walking by the biped walking machine. In Morecki, A., Bianchi, G., and Kedzior, K., editors, *Fourth Symposium on Theory and Practice of Walking Robots*, pages 341–351. Polish Scientific Publishers.

Kelbley, R. J. (2006). Guidance and control of an unmanned surface vehicle. Master's thesis, Computer Engineering, University of California, Santa Cruz.

Kelly, R. (1993). Comments on adaptive pd controller for robot manipulators. *IEEE Trans. Robot. Automat.*, 9:117–119.

Kelly, R. (1997). Pd control with desired gravity compensation of robotic manipulators: A review. *Int. J. Robot. Res.*, 16(5):660–672.

Khalil, W. and Dombre, E. (2004). *Modeling, Identification and Control of Robots*. Kogan Page Science.

Kim, D., Kim, N.-H., Seo, S.-J., and Park, G.-T. (2004). *Fuzzy Modeling of Zero Moment Point Trajectory for a Biped Walking Robot*. Lecture Notes in Computer Science. Springer-Verlag GmbH, 3214 edition.

Kim, J. O., Lee, B. R., Chung, C. H., Hwang, J., and Lee, W. (2003). *The Inductive Inverse Kinematics Algorithm to Manipulate the Posture of an Articulated Body*. Lecture Notes in Computer Science. Springer-Verlag GmbH, 2657 edition.

Laib, A. (2000). Adaptive output regulation of robot manipulators under actuator constraints. *IEEE Trans. Robot. Automat.*, 16:29–35.

Larimore, W. E. (1990). Canonical variate analysis in identification, filtering and adaptive control. In *IEEE Conference on Decision and Control*, pages 596–604.

Lim, H. and Takanishi, A. (2005). Compensatory motion control for a biped walking robot. *Robotica*, 23(01):1–11.

Lin, W., Qin, S. J., and Ljung, L. (2004). A framework for closed-loop subspace identification with innovation estimation. Technical Report 2004-07, Department of Chemical Engineering, The University of Texas at Austin, Austin, TX 78712, USA and Linköping University, SE-581 83 Linköping, Sweden.

Ljung, L. (1999). *System identification: theory for the user*. Prentice-Hall, Upper Saddle River, NJ, USA.

Lohar, F. A. (2000). $\mathscr{H}_\infty$ and μ-synthesis for full control of helicopter in hover. In *38th Aerospace Sciences Meeting and Exhibit*, Reno, NV. American Institute of Aeronautics and Astronautics.

Lope, J. d., Zarraonandia, T., González-Careaga, R., and Maravall, D. (2003). Solving the inverse kinematics in humanoid robots: A neural approach. *Lecture Notes in Computer Science*, 2687 / 2003:177–184.

McGeer, T. (1990). Passive dynamic walking. *The Internatinal Journal of Robotics Research*, 9(2):62–82.

McMahon, T. (1984). Mechanics of locomotion. *The International Journal of Robotics Research*, 3(2):4–28.

Morari, M. and Lee, J. H. (1999). Model predictive control: Past, present and future. *Computers and Chemical Engineering*, 23:667–682.

Norton, J. P. (1986). *Introduction to Identification*. Academic Press.

Ogihara, N. and Yamazaki, N. (2001). Generation of human bipedal locomotion by a bio-mimetic neuro-musculoskeletal model. *Biological Cybernetics*, 84:1.

Overschee, P. V. and Moor, B. D. (1994). N4SID: Subspace algorithms for the identification of combined deterministic-stochastic systems. *Automatica*, 30(1):75–93.

Overschee, P. V. and Moor, B. D. (1996). *Subspace Identificiation for Linear Systems*. Kluwer Academic Publishers.

Paul B. Brugarolas, M. G. S. (2004). Learning about dynamical systems via unfalsification of hypotheses. *International Journal of Robust and Nonlinear Control*, 14(11):933–943.

Pratt, J. and Pratt, G. (1998). Intuitive control of a planar bipedal walking robot. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.

Pratt, J. and Pratt, G. (1999). Exploiting natural dynamics in the control of a 3D bipedal walking simulation. In *Proceedings of the International Conference on Climbing and Walking Robots (CLAWAR)*.

Pratt, J. E. (2000). *Exploiting Inherent Robustness and Natural Dynamics in the Control of Bipedal Walking Robots*. PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology.

Raibert, M. (1986). *Legged Robots That Balance*. The MIT Press.

Sakagami, Y., Watanabe, R., Aoyama, C., Matsunaga, S., Higaki, N., and Fujimura, K. (2002). The intelligent ASIMO: system overview and integration. In *IEEE/RSJ International Conference on Intelligent Robots and System*, volume 3, pages 2478–2483.

Sapio, V. D. and Khatib, O. (2005). Operational space control of multibody systems with explicit holonomic constraints. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*.

Schwind, W. J. (1998). *Spring Loaded Inverted Pendulum Running: A Plant Model*. PhD thesis, University of Michigan.

Sciavicco, L. and Siciliano, B. (2000). *Modelling and Control of Robot Manipulators*. Springer, 2nd edition.

Staib, W. E. and Staib, R. R. (1992). A neural network electrode positioning optimization system for the electric arc furnace. In *International Joint Conference on Neural Networks*, volume 111, pages 1–9.

Taga, G. (1995). A model of the neuro-musculo-skeletal system for human locomotion. I. Emergence of basic gait. *Biological Cybernetics*, 73(2):97–111.

Takegaki, M. and Arimoto, S. (1981). A new feedback method for dynamic control of manipulators. *ASME J. Dyn. Syst., Meas., Control*, 102:119–125.

Tischler, M. B., Driscoll, J. T., Cauffman, M. G., and Freedman, C. J. (1994). Study of bearingless main rotor dynamics from frequency-response wind tunnel test data.

In *American Helicopter Society Aeromechanics Specialists Conference*.

Tomei, P. (1991). Adaptive pd controller for robot manipulators. *IEEE Trans. Robot. Automat.*, 7:565–570.

Verhaegen, M. and Dewilde, P. (1992). Subspace model identification. part i: the output-error state-space model identification class of algorithms. *International Journal of Control*, 56:1187–1210.

Wang, J. and Qin, S. J. (2004). A new deterministic-stochastic subspace identification method using principal component analysis. Technical report, Department of Chemical Engineering, The University of Texas at Austin.

Wang, J. M.-C. (2005). Gaussian process dynamical models for human motion. Master's thesis, Graduate Department of Computer Science, University of Toronto.

Wen, J., Kreutz-Delgado, K., and Bayard, D. (1992). Lyapunov function-based control laws for revolute robot arms. *IEEE Trans. Automat. Contr.*, 37:231–237.

Wernholt, E. (2004). *On Multivariable and Nonlinear Identification of Industrial Robots*. PhD thesis, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden.

Wilson, E. (1997). Adaptive profile optimization for the electric arc furnace. In *Steel Technology International*, pages 140–145.

Wolkotte, P. T. (2003). Modelling human locomotion. Technical report, Institute of Electronic Systems, Aalborg University.

Woodley, B., How, J., and Kosut, R. (2001a). Model free subspace based $\mathscr{H}_\infty$ control. In *Proceedings of the 2001 American Control Conference*, volume 4, pages 2712–2717.

Woodley, B., Kosut, R., and How, J. (1998). Uncertainty model unfalsification with simulation. In *Proceedings of the 1998 American Control Conference*, volume 5, pages 2754–2755.

Woodley, B. R. (2001). *Model free subspace based $\mathscr{H}_\infty$ control*. PhD thesis, Department of Electrical Engineering, Stanford University.

Woodley, B. R., How, J. P., and Kosut, R. L. (2001b). Subspace based direct adaptive $\mathscr{H}_\infty$ control. *International Journal of Adaptive Control and Signal Processing*, 15(5):535–561.

Xie, M. (2003). *Fundamentals of Robotics*, volume 54 of *Machine perception and artificial intelligence*. World Scientific.

Zhang, Y., Tian, H., Wang, Q., and Qiang, W. (2000). Servo control in joint space of biped robot using nonlinear $\mathscr{H}_\infty$ strategy. In Jiang, D. and Wang, A., editors, *Proceedings of SPIE, International Conference on Sensors and Control Techniques (ICSC 2000)*, volume 4077, pages 386–391.

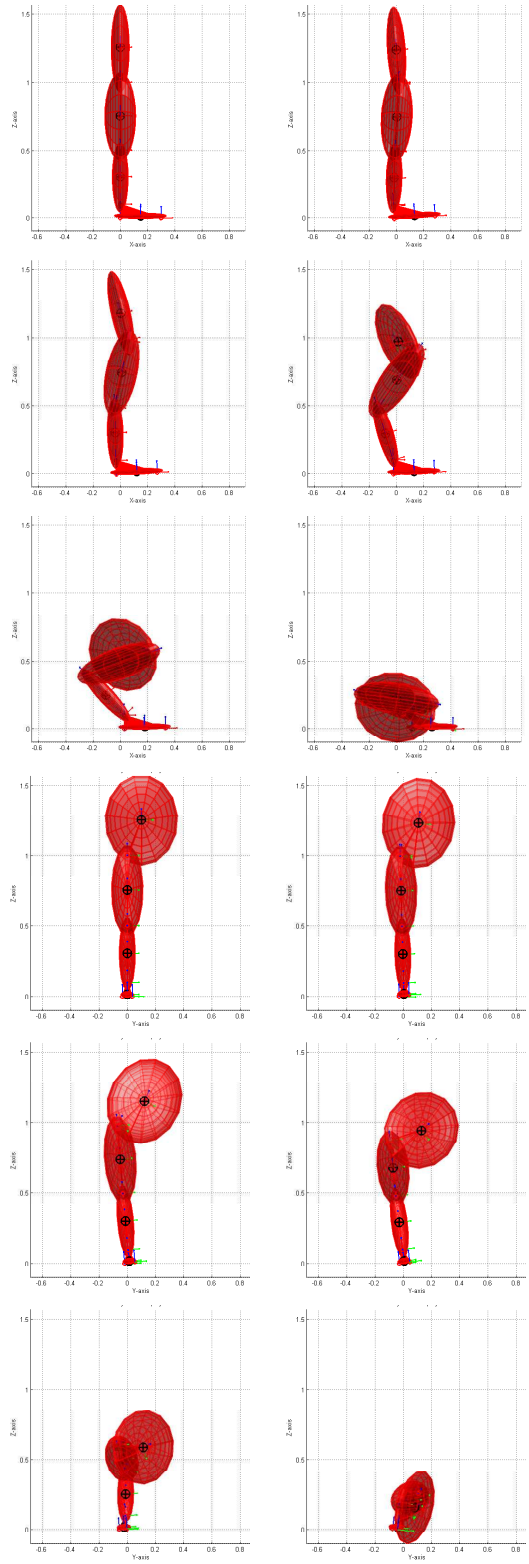


Figure 4: Free fall of a biped leg with exogenous force signals at its joints. Top six and bottoms ones are shots taken from the same simulation but from different angles after every 0.1 seconds.

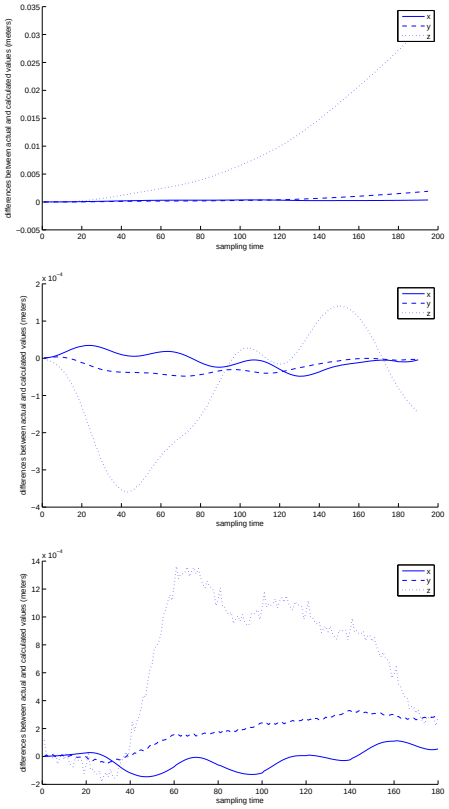


Figure 5: Error in the calculation of torso position. Above graphs are with $i = 5, i = 10$, and $i = 20$ respectively. Note that the largest movement of torso is in the z-direction, the error is also the most in this direction.

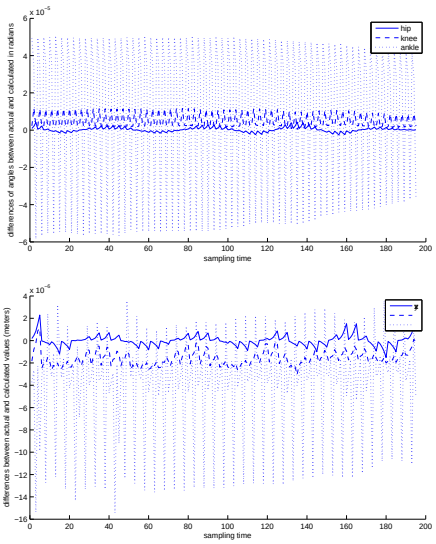


Figure 6: Error in the calculation of torso position when data is updated from actual torso position after every prediction. Above graphs are with $i = 5$ and $i = 20$ respectively. Note that even for very small prediction horizon *i.e.* $i = 5$, the error is very small.

SCHEDULING OF MULTI-PRODUCT BATCH PLANTS USING REACHABILITY ANALYSIS OF TIMED AUTOMATA MODELS

Subanatarajan Subbiah, Sebastian Panek, Sebastian Engell
Process Control Laboratory (BCI-AST), University of Dortmund, 44221 Dortmund, Germany
{*subbiahs, paneks, engells*}*@bcinet.bci.uni-dortmund.de*

Olaf Stursberg
Industrial Automation Systems (EI-LSR), Technical University of Muenchen, Muenchen, Germany
stursberg@tum.de

Keywords: Scheduling, timed automata, reachability analysis, multi-product batch plant.

Abstract: Standard scheduling approaches in process industries are often based on algebraic problem formulations solved as MI(N)LP optimization problems to derive production schedules. To handle such problems techniques based on timed automata have emerged recently. This contribution reports on a successful application of a new modeling scheme to formulate scheduling problems in process industries as timed automata (TA) models and describes the solution technique to obtain schedules using symbolic reachability analysis. First, the jobs, resources and additional constraints are modeled as sets of synchronized timed automata. Then, the individual automata are composed by parallel composition to form a global automaton which has an initial location where no jobs have been started and at least one target location where all jobs have been finished. A cost optimal symbolic reachability analysis is performed on the composed automaton to derive schedules. The main advantage of this approach over other MILP techniques is the intuitive graphical and modular modeling and the ability to compute better solutions within reasonable computation time. This is illustrated by a case study.

1 INTRODUCTION

Scheduling has applications in many fields and one among them are the chemical industries employed with multi-product batch plants. Multi-product batch plants in contrast to continuous plants offer the advantage of an increased flexibility with respect to the variation of the products, the volume of production, and the range of recipes that can be processed by the equipment. This flexibility makes the scheduling task challenging. The scheduler has to take a large number of decisions to answer questions of what, when, where and how to produce the products in order to satisfy the market demand. The scheduling task is particularly challenging due to numerous constraints arising from process topology, the interlinks between pieces of equipment, inventory policies, material transfers, batch sizes, batch processing times, demand patterns, changeover procedures, resource and time constraints, cost functions and degrees of uncertainty. Many scientists in the process logistics community have contributed approaches to model and solve such problems (Kallrath, 2002). In most prominent approaches the problem is specified using the frameworks of State Task Networks (STN) or Resource Task Networks (RTN) (Pantelides, 1994), further transformed to an optimization problem modeled as a mixed integer program (MILP or MINLP) and solved using state-of-the art solvers such as CPLEX, XPRESS-MP, etc. The modeling of mathematical optimization problems is generally cumbersome, tedious and requires not only experience in algebraic modeling but also a deep knowledge of the solver and its internal algorithms.

Recently, techniques based on reachability analysis for timed automata to solve scheduling problems have gained attention. The framework of TA has been originally proposed by (Alur and Dill, 1994) to model timed systems with discrete dynamics. Many powerful tools for the modeling and analysis of real-time systems such as UPPAAL, IF, KRONOS and TAOPT have emerged. The authors of (Abdeddaim and Maler, 2001) have proposed to use TA to solve job-shop scheduling problems based on the symbolic reacha-

bility analysis of TA. Promising results on challenging benchmark instances in job-shops were reported in (Panek et al., 2006). The particular appeal of this approach comes from the modular and partly graphical modeling which enables inexperienced users to build models. Another advantage is the availability of powerful search algorithms that can easily be modified and extended for special purposes. Similar to other approaches the reachability analysis of TA also suffers from the drastic increase in the computational cost with an increase in the problem size. The problem of state-space explosion in TA models can be reduced to some extent by state-space reduction techniques (Panek et al., 2007), but arises again when the problem instance is scaled. An advantage of the TA-based approach is that good, though not provable optimal solutions are often found relatively quickly.

The structure of the rest of this paper is as follows: In Section 2, Resource Task Networks (RTN) are introduced as the basic model for specifying batch scheduling problems. Section 3 explains how the RTN model can be translated into TA, and how the corresponding scheduling problem can be solved by symbolic reachability analysis. In section 4, different greedy strategies to reduce the solution space are explained. The description of the case study and the results of the experiments for the example considered are provided in section 5. The paper closes with conclusions and an outlook.

2 MODELING WITH RTN

In the context of mathematical programming, STN and RTN serve as starting points for formulating integrated mathematical models that can be solved to obtain production schedules. A short description of the RTN representation is given in this section based upon a small example: a two stage process with 3 tasks and 3 resources. The resources R_1, R_2 and R_3 handle batch sizes of b_1, b_2 and b_3 units. In the first stage, the single raw material S_1 is processed by task T_1 in resource R_1 to produce the intermediate material S_2. In the second stage, the intermediate material S_2 is processed by tasks T_2 in R_2 and T_3 in R_3 to produce the end products S_3 and S_4. The tasks T_1, T_2 and T_3 requires d_1, d_2 and d_3 time units, respectively. There exists a timing constraint between tasks T_1 and T_3 such that T_3 must be started within Y time units after T_1 has finished, as the intermediate product S_2 is unstable and becomes unusable after Y time units. We assume in the example that dedicated storage tanks are available for each of the products that have the same names as the products. The RTN model consists of four basic elements; (i) *state nodes*, to represent the feeds, intermediate materials and the final products, (ii) *task nodes* to represent the process operations that transforms one or more input states to one or more output states, (iii) *resource nodes* to represent the production units/equipments, and (iv) *arcs* with fractional values to indicate the flow and the percentage of material transported to the successor state. The default values for the flow are 1. The RTN description of the example process is shown in Figure 1. The feed, intermediate materials and products (S_1,$\cdots$ S_4) are represented as circles with diagonal lines. The tasks T_1, T_2 and T_3 are represented as rectangles and connected to the corresponding resources R_1, R_2 and R_3, respectively by dashed lines.

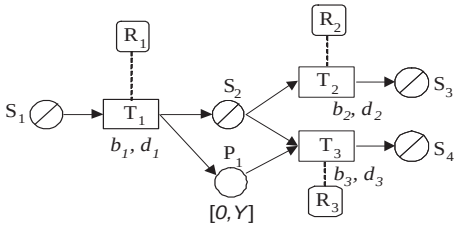


Figure 1: RTN description of the example process.

The standard RTN framework offers only limited possibilities to describe timing constraints between tasks. The timing constraint between task T_1 and T_3 is modeled by a new type of node called *place* denoted by P_1. The notion of places is motivated by *Timed and Hybrid Petri Nets* (Barthomieu and Diaz, 1991). Places can be connected to tasks in the same fashion as states, however unlike states they represent no physical products. They contain either one or no *token*. When one of the predecessor operations is finished, a token is assigned in the place and it is consumed when one of the successor operation starts. The place in the RTN is marked by the time interval $[0, Y]$, signifying that a token can stay in the place for at least 0, and for at most Y time units.

2.1 Timed Automata

Timed automata (TA) are extensions of finite-state automata with clocks and are used to model and analyze systems with discrete dynamics and timed behaviour. Time in TA is modeled using a set of clocks. A short and informal definition of TA is given below, for a detailed definition for TA please refer to (Alur and Dill, 1994).

2.1.1 Syntax

A timed automata is defined by a tuple, TA = (L, l_o, F, $\mathbb{C}$, E, inv), in which L represents the finite set of

locations, l_0 represents the initial location and F represents the set of final locations. $\mathbb{C}$ represents the set of clocks assigned to the TA. The set $E \subset L \times \Phi(\mathbb{C}) \times Act \times \mathcal{P}(\mathbb{C}) \times L$ represents the transitions. $\Phi(\mathbb{C})$ are guards specified as conjunctions of constraints of the form $c_i \otimes n$ or c_i - $c_j \otimes n$, where c_i, $c_j \in \mathbb{C}$, $\otimes \in \{ <, \leq, ==, \neq, >, \geq \}$ and $n \in \mathbb{N}$. *Act* denotes a set of actions (e.g. invoking a new event or changing the value of a variable). $\mathcal{P}(\mathbb{C})$ represent the power set over elements of $\mathbb{C}$. *inv* represents a set of invariants which assign conditions for staying in locations. An invariant must evaluate to true when the corresponding location is active. The automaton is forced to leave the location when the invariant evaluates to false.

(l, g, a, r, l') denotes a transition between a source location l and target location l' with a guard $g \in \Phi(\mathbb{C})$, performing an action $a \in Act$ and resetting the clocks $r \in \mathcal{P}(\mathbb{C})$. Such a transition can occur only when the transition guard $g \in \Phi(\mathbb{C})$ is satisfied.

Figure 2 shows a simple TA with three locations l_1, l_2 and l_3 connected by transitions. The clock c is reset in the first transition. The invariant $c \leq 3$ in location l_2 refers to the clock c and enforces the following transition which is enabled by the guard $c \geq 2$. The transitions are labeled with actions α and ϕ, respectively. In the context of scheduling we use the transitions α for allocating a resource to a task and ϕ for releasing a resource of the task after completion.

TA models are created in a modular fashion as sets of interacting automata. The interaction between TA is established by synchronized transitions. Two transitions are synchronized when they have the same synchronization labels from *Act*. The corresponding automata change their locations simultaneously. The parallel composition of a pair of TA is performed by combining a set of TA into one composed global TA using the synchronization labels.

The parallel composition of two individual automata $A_1 = (L_1, l_{1,0}, F_1, \mathbb{C}_1, E_1, inv_1)$ and $A_2 = (L_2, l_{2,0}, F_2, \mathbb{C}_2, E_2, inv_2)$ is given by $A_{comp} = A_1||A_2 = (L_1 \times L_2, (l_{1,0}, l_{2,0}), F_1 \times F_2, \mathbb{C}_1 \cup \mathbb{C}_2, E, inv)$, with $l = (l_1, l_2)$ and $E = E_1(l_1) \wedge E_2(l_2)$. A transition in the composed automaton A_{comp} $(l, g, a, r, l') \in E$ exist iff there exist transitions $(l_1, g_1, a_1, r_1, l'_1) \in E_1$ in A_1 and $(l_2, g_2, a_2, r_2, l'_2) \in E_2$ in A_2, with guard g = $g_1 \wedge g_2$ satisfied, $r = r_1 \cup r_2$ and $a \in ((Act_1 \cup \{0\}) \times (Act_2 \cup \{0\}))$. The symbol 0 here denotes an empty action in which no state transition occurs. Similarly n individual TA can be composed step by step to obtain the composed automaton $A_{comp} := A_1||A_2||\cdots||A_n$.

An extension of TA with the notion of costs are *priced* TA (PTA) (Behrmann et al., 2001). A priced TA is equipped with an additional function P: $L \cup E \rightarrow \mathbb{R}^{\geq 0}$ which assigns a cost to transitions and cost rates to locations. The cost of staying in a location for d time units is given by $d \cdot P(L)$.

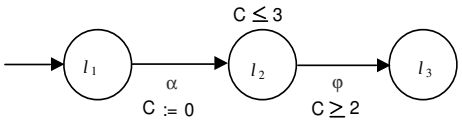


Figure 2: A simple timed automata.

2.1.2 Semantics

The semantics of a composed PTA A is defined in terms of a labeled transition system $(\mathbb{Q}, (l_0, u_0, 0), \Delta)$, with a state space $\mathbb{Q} \subseteq L \times \mathbb{R}^{|\mathbb{C}|} \times R^{\geq 0}$, the initial state given by $(l_0, u_0, 0) \in \mathbb{Q}$ with cost 0 and Δ representing the infinite transition relation. The state is a tuple (l, u, p), where l denotes the location, u denotes a vector of clock valuations and p represents the cost. Δ contains the following transitions:

1. Time transitions: $(l, u, p) \xrightarrow{\tau} (l, u + \mathbf{1}\tau, p + P(L) \cdot \tau)$ iff for every $\theta \in [0, \tau]$ the invariant $inv(l)$ evaluates to true. The vector $\mathbf{1}$ is composed of ones and has the dimension equal to the number of clocks in the set $\mathbb{C}$.

2. Discrete transitions: $(l, u, p) \xrightarrow{a} (l', u', p + P(e))$ where there exists a transition $e = (l, g, a, r, l')$ in E and $\theta \geq 0$ such that $g(u + \mathbf{1}\theta)$ is satisfied and $u' = (u'_1, u'_2, \cdots, u'_{|\mathbb{C}|})^T$ with $u'_i = 0$ for every $c_i \in r$ and $u'_i = u_i + \theta$ otherwise.

A sequence of states and state transitions from Δ constitutes a trace ρ of the composed priced automaton A given by $\rho = (l_0, u_0, 0) \xrightarrow{\delta_1} (l_1, u_1, p_1) \xrightarrow{\delta_2} \cdots$ with $\delta_i \in \Delta$. The state transitions in the trace are alternating time transitions and discrete transitions. Each trace corresponds to runs of the individual automata thereby characterizing a possible evolution of the system. An optimal trace $\rho*$ is a path from the initial state (l_0, u_0, 0) to a target state (l_n, u_n, p_n) with minimal p_n. The set of traces is infinite due to the fact that time delays τ are elements of time intervals defined by invariant and guard constraints. This then makes computing the optimal run from all possible runs a challenging task.

A solution to this problem is provided by the *zone abstraction* technique. A zone Z is an infinite set of possible clock valuations in a location l, which is expressed as a conjunction of finitely many inequalities such as c_i - $c_j \leq n$ or $c_i \leq n$ with c_i, $c_j \in \mathbb{C}$ and $n \in \mathbb{R}^{\geq 0}$. A symbolic state is formed by the combination of a location and the zone for the corresponding location (l, z). All symbolic states q^s constitute the symbolic state space Q_s. The symbolic semantics is defined in terms of a symbolic transition system in

which zones Z replace single clock valuations u. A symbolic trace ρ^s is a sequence of symbolic states in which the zones are computed by applying intersections, delay and reset operations defined for zones.

The main advantage of the zone abstraction is that the symbolic state space Q_s is enumerable and forms a *directed symbolic reachability graph* with nodes from Q_s and arcs from Δ_s. The aim of the cost optimal reachability analysis is to find the cheapest path from the initial state to a target symbolic state using weighted shortest path algorithms. The resulting shortest path obtained corresponds to the optimal symbolic trace ρ^{s*}. The shortest trace ρ^* can then be derived by picking the minimal clock valuations u_i^* from the zones Z_i^* of the symbolic states along the path.

3 SCHEDULING BASED ON TA

After the global composed timed automaton is constructed through *parallel composition*, a (cost-optimal) reachability analysis is performed to derive the schedules. This enumerative technique explores the symbolic state space step-by-step by evaluating the symbolic successor relation on-the-fly. The basic reachability algorithm from (Panek et al., 2007) is given below:

Algorithm 1 *A cost-optimal reachability algorithm*

```
1   cost* = ∞
2   W := {q0}; P := ∅
3   While W ≠ ∅
4       q = selectRemove(W)
5       If q ∉ P
6           P := P ∪ {q}
7           If cost(q) < cost*
8               If final(q)
9                   cost* := cost(q)
10              Else
11                  S := succ(q)
12                  S' := reduce(S)
13                  W := W ∪ S'
14              End
15          End
16      End
17  End
```

The algorithm operates on the nodes of the graph which contains the information on the location, clock valuations, accumulated cost and some additional data. The list of nodes that are to be explored is represented by the set $\mathcal{W}$ called the *waiting list*. The waiting list $\mathcal{W}$ initially contains only the initial node q_0 which in our case corresponds to the state where no jobs have been started. The set $\mathcal{P}$ represents the *passed list*, a list of already explored nodes, which is normally empty at the start of the search algorithm. $cost^*$ holds the cost of the best path found so far and q corresponds to the node that is currently explored. $\mathcal{S}$ is the set of immediate successors that can be reached by a single transition from q. The function $selectRemove(\mathcal{W})$ selects and removes a node from $\mathcal{W}$ which is minimal according to the ordering $\leq_{sel}$ defined by the tree search strategy. In order to avoid visiting and exploring the same node repeatedly, the test $q \notin \mathcal{P}$ is performed. $final(q)$ checks whether q is a target node. The function $succ(q)$ computes the immediate successor nodes of q and stores them temporarily in $\mathcal{S}$. The function *reduce* ($\mathcal{S}$) is used to remove those nodes that cannot contribute to a better path or to the optimal run. The test $cost(q) < cost^*$ compares the cost value of the current node with the best cost value found, thereby avoiding exploration of non-optimal runs if the comparison fails. The decision criteria for the function *reduce* are based on timed transitions to successor nodes and will be discussed in detail in the following sections.

4 REDUCTION TECHNIQUES

This section discusses three methods to make the search for the optimal schedule more efficient.

4.1 Immediate Schedules

In (Abdeddaim and Maler, 2001), it has been shown that for job-shop problems with the makespan minimization as objective, the trace corresponding to an optimal schedule is always *immediate*. A non-immediate trace contains a transition sequence $\cdots \rightarrow (l,u) \xrightarrow{\tau} (l,u+\tau) \xrightarrow{\alpha} (l',u') \rightarrow \cdots$ where the α transition taken at $(u+\tau)$ is already enabled at $(u+\tau')$ with $\tau' < \tau$. Such traces exhibit periods of useless waiting in the schedule. Waiting is useless if, e.g., no tasks are started while the required resource is available (see Figure 3). Non-immediate traces can be transformed to immediate traces by reducing such waiting periods to zero. A reduction by $\tau - \tau'$ leads to the partial trace $(l,u) \xrightarrow{\tau'} (l,u+\tau') \xrightarrow{\alpha} (l',u')$ and can reduce the overall length of the corresponding schedule. This infers that it is sufficient to search for immediate traces in the symbolic state space. The effect of the reductions of the waiting times is that the zones in the symbolic states collapse to single clock valuations and the symbolic reachability graph is reduced to nodes representing symbolic states (l,u). The arcs

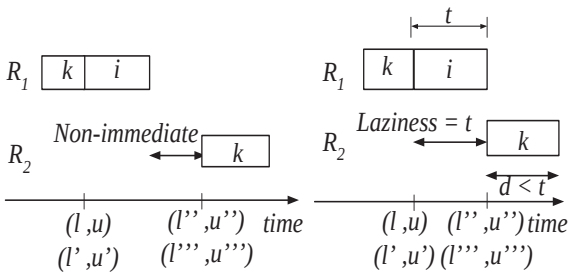


Figure 3: Non-immediate and Lazy schedule.

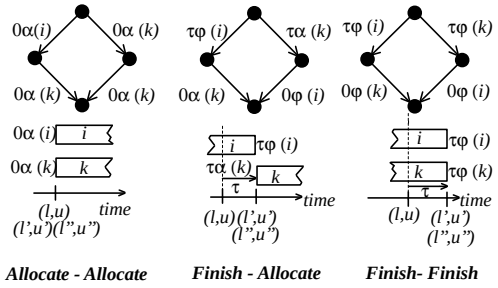


Figure 4: Sleep set configurations.

of the symbolic reachability graph become timed transitions: pairs of discrete and time transitions τa in which τ is the smallest possible delay needed to satisfy the guard of the discrete transition with the action a. Zones Z in the symbolic states can be replaced by vectors of clock valuations u such that (l,u) is a state of an immediate trace.

4.2 Ordinary and Strict Non-Laziness

In (Abdeddaim and Maler, 2001) also a concept to avoid unnecessary waiting known as *non-laziness* was introduced. Consider a case in which a current node (l,u) with two discrete transitions as successors namely $\alpha_i^{R_1}$ and $\alpha_k^{R_2}$ that allocate resources R_1 and R_2 to jobs i and k, exists. The releasing transitions are represented as $\phi_i^{R_1}$ and $\phi_k^{R_2}$. The transition $\alpha_k^{R_2}$ was already enabled at the same time when the transition $\alpha_i^{R_1}$ was enabled. In the partial trace $(l,u) \overset{0\alpha_i^{R_1}}{\rightarrow} (l',u') \overset{t\phi_i^{R_1}}{\rightarrow} (l'',u'') \overset{0\alpha_k^{R_2}}{\rightarrow} (l''',u''')$ the enabled transition $\alpha_k^{R_2}$ is taken at a later stage by waiting to finish job i thereby introducing laziness. In order to remove laziness in a trace, waiting in a node for t or more than t units is forbidden when all the following conditions are satisfied:

(a) there exists a successor of (l,u) in which an operation of duration d can be started by an enabled discrete transition that allocates a resource to a job.

(b) The duration d of this operation satisfies $d \leq t$

(c) if the allocating transition is permanently enabled and the resource allocated by that transition is available for more than $d-t$ units (provided $d > t$).

The term *ordinary laziness* is used since a short waiting time according to conditions (b) and (c) is permitted. A stricter condition with respect to waiting is the *strict non-laziness*, which is obtained by removing conditions (b) and (c). Thus, whenever there exists at least one timed transition to start a new task waiting is forbidden in a state. In comparison to ordinary non-laziness this scheme prunes more time successors. However the strict condition might also prune the optimal trace as in some problems the optimal schedules may have a short waiting period.

4.3 Sleep-Set Method

The sleep-set method was originally introduced in (Godefroid, 1991) to reduce possible partial orders and to avoid exploring obsolete runs in the reachability graph. The application of the method to job-shop problems was first implemented in (Panek et al., 2006). In the symbolic reachability graph different symbolic traces which correspond to the same schedule may exist. This is due to the *interleaving semantics* of the TA model. Such traces includes transitions that are taken at the same absolute time but can be traversed in different logical orders. In general, such transitions share a common symbolic state at the start and at the end of the transition with the same cost. Three common and possible configurations that can be identified in the reachability graph are shown in Figure 4. The left part of the figure shows a pair of α transitions that can be traversed in two different ways from a common start node (l,u). The possible logical order of the transition pair are $\{\alpha(i),\alpha(k)\}$ and $\{\alpha(k),\alpha(i)\}$. Both allocating transitions share the same end state, cost and corresponds to the same schedule shown. Such traces form so called *diamond structures* in the reachability graph. Similar kind of structures arising due to pairs of α and ϕ transitions and the corresponding schedule are shown in the same figure. Thus, it is evident that only one of the traces in the diamond structure has to be explored and the rest can be pruned.

The idea of the sleep-set method to remove obsolete traces is to compare pairs of immediate successor transitions from a node and mark some of them as candidates to be pruned. However pruning of marked candidates can only take place when both transitions are independent. Pairs of timed transitions are said to be independent if the corresponding operations do not require the same resource and taking one transition does not block the other. It should be noted that

pairs of ϕ and mixed pairs of α and ϕ transitions are independent as a single resource cannot finish two operations at the same time. Not all pairs of α transitions are always independent as they might compete for the same resource. An outgoing transition $\tau\delta$ is said to be time persistent in a given symbolic state (l,u,p) at time t, if the discrete transition δ is taken at $t+\tau$ for every partial run starting from (l,u,p) with $\tau \geq 0$. Given a symbolic state with outgoing transitions $\tau\delta_i$ and $\tau\delta_j$, $\tau\delta_j$ is marked as a candidate for reduction if $\tau\delta_i$ is time persistent and vice versa. If both are time persistent, then the transition with a higher resource index is removed. This reduction however does not work when comparing two α transitions as they are not time persistent. An allocated task must always be finished. This shows that all outgoing transitions that release a resource are time-persistent and appear in future parts of the trace. Since α transitions are not time persistent, in general they cannot be removed. However it is possible to identify *eager* α transitions which become implicitly time persistent in a non-lazy run. A transition $\tau\alpha_i$ which is an immediate successor $\mathcal{S}$ of a state is called eager if for all $\tau_j\phi_j \in \mathcal{S}$ either (a) $\tau + dur(\alpha_i) \leq min_j\{\tau_j|\tau_j\phi_j \in \mathcal{S}\}$ or (b) $\tau \geq max_j\{\tau_j|\tau_j\phi_j\}$ or (c) $\tau \geq 0$ if no ϕ transitions exist in $\mathcal{S}$. A strict approach is to use the same strategy to remove transitions if they are independent and not to check whether the transition is time persistent. This strict pruning strategy causes a considerable reduction of the reachability graph to a considerable amount. However, it may remove the optimal trace as a pruned transition can be part of the optimal trace.

5 EXAMPLE PROCESS

In order to depict the applicability and the effectiveness of the approach discussed here, the case study presented by (Bauer et al., 2000) is considered. The case study is a multi-product chemical batch plant with 3 stages in which two end-products, ***X*** and ***Y***, are produced from three raw materials, ***A***, ***B*** and ***C***. The production process considered is: one batch of material ***A*** and ***C*** each reacts to produce one batch of product ***X***; similarly one batch of ***B*** and ***C*** reacts to produce one batch of product ***Y***. The plant executing the production process is shown in Figure 5.

The first stage consists of three buffer tanks B11, B12 and B13 that store the raw materials ***A, C*** and ***B***. The second stage consists of three reactors R1, R2 and R3. Each reactor may be filled from each raw material buffer tank in the first stage; implying that it is possible to produce either product ***X*** or ***Y*** in each reactor. After processing the materials a reactor may

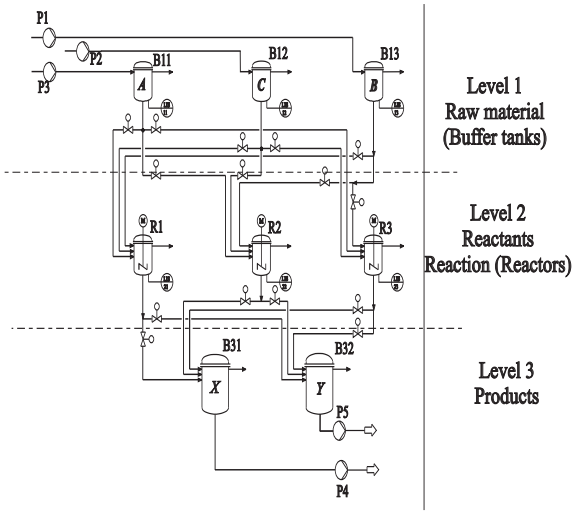


Figure 5: The structure of the example process.

contain one batch of the product (resulting from two batches of raw material). The third stage consists of two buffer tanks B31 and B32 which are used to store the end-products ***X*** and ***Y***. Each of the tanks has a maximum capacity to store three batches of the product. The proposed scheduling problem is to derive production schedules to produce 2 batches of the end-products with a minimum makespan. After being processed by the reactors the materials must be drained into the buffer tanks in the third stage immediately.

RTN description of the problem considered is shown in Figure 6. Materials ***A, C*** and ***B*** are represented as S_1, S_2 and S_3. The tasks T_1, T_2 and T_3 represents the task of pumping materials ***A, C*** and ***B*** into buffer tanks B11, B12 and B13, respectively. The tasks T_4, T_6 and T_8 represent the step of draining one batch of ***A*** from B11 into the reactors R1, R2 and R3. Similarly, tasks T_{11}, T_{13} and T_{15} represent draining of material ***B*** from B13 into the reactors. The tasks T_5, T_7 and T_9 represent mixing one batch of ***C*** from B12 with one batch of ***A***, already present in R1, R2 and R3 in order to produce ***X***. The tasks T_{10}, T_{12} and T_{14} represent similar actions for producing one batch of ***Y***. The tasks T_{16}, T_{17} and T_{18} represent the step of draining one batch of ***X*** from R1, R2 and R3, respectively, into the buffer tank B31. Tasks T_{19}, T_{20} and T_{21} represent draining one batch of ***Y*** from R1, R2 and R3 into tank B32. The tasks T_{22} and T_{23} represent draining of two batches of ***X*** from B31 and two batches of ***Y*** from B32 to the product storage, respectively. The constraint on the waiting time is expressed by the *places* P_2, P_4, P_6, P_8, P_{10} and P_{12}. The *places* P_1, P_3, P_5, P_7, P_9 and P_{11} are established to ensure that material ***C*** is drained into the reactors only after either ***A*** or ***B*** has been drained.

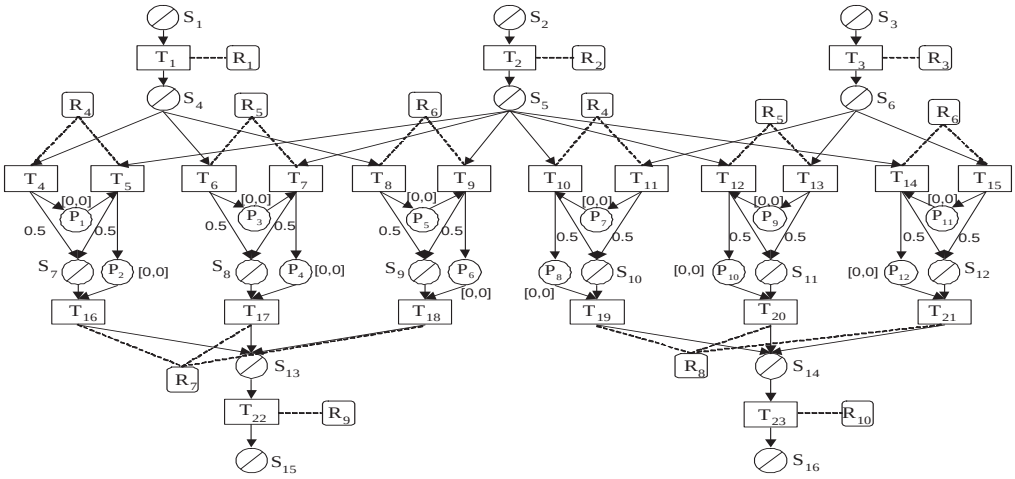


Figure 6: The RTN of the example process.

5.1 TA Model

The TA model consists of 10 resource automata to model the individual resources. The places are modeled as separate automata. The resource automata and the place automata have one clock each to measure the task durations and time interval for storage. An additional global clock which is never reset is included in the model to measure the makespan. The resource automaton for resource R_4 and the place automaton P_2 are shown in Figure 7. The automaton R_4 has one idle location and four busy locations indicating the execution of either of the tasks T_4 or T_5 or T_{10} or T_{11}. Tasks T_4 and T_{11} process one batch of material S_4 and S_6 with task duration of 15 and 11 units. The timing constraint between task T_5 and task T_{16} is modeled by the place automaton P_2 on the right side. Task T_5 after finishing its process in R_4 produces a token in the place P_1 by taking the synchronized transitions in R_5 and P_2 with the label ϕ_5. The clock C_{12} is reset and forces P_2 to leave the location *filled* without waiting. This happens by starting task T_{16} and by taking the synchronized transitions with label α_{16} in R_7 and P_2. An automatic translation from the RTN to the TA model was performed by the tool TAOPT (Panek et al., 2005).

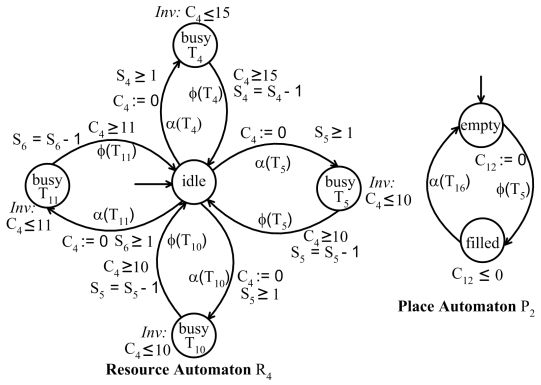


Figure 7: Structure of the resource and place automaton.

5.2 Computational Experiments

The scheduling problem presented in the example process was solved by a Xeon machine with a CPU speed of 3.06 GHz, 2.0 GB of memory and a linux operating system. For the cost optimal symbolic reachability analysis, the software tool TAOPT (Panek et al., 2005) which supports different state space reduction techniques as discussed above: immediate traces, (strict) *non-lazy runs* and *sleep-set method* was used. A combination of the depth-first and best-first strategy was used for exploring the graph. Among the nodes in the waiting list, the node with the maximum depth is given a higher priority, among the nodes with the same depth, priority is given to the node with a minimum cost. If two or more nodes have the same depth and cost, one of them is chosen randomly.

The Table 1 shows the results obtained by restricting the search to only immediate traces represented as *None* and with other state space reduction techniques: ordinary non-lazy runs ($NL_{ordinary}$), strict non-lazy runs and strict sleep-set method (SSM_{strict}). For all tests the complete reachability graph was explored. The second column that represents the total time needed to compute the reachability graph and the total number of nodes explored in the the third column reveal that the strict sleep-set method reduces the state space and computational effort considerably. On the other hand it prunes the optimal solutions. The fourth column represents the time taken in CPU seconds to

Table 1: Computational results.

			First solution			Best solution
Reduction technique	Total time	Total nodes	time	nodes	C_{max}	C_{max}
None	58.832	493167	0.032	1875	100	98
$NL_{ordinary}$	2.104	37483	0.048	2157	98	98
SSM_{strict}	0.476	13374	0.120	6492	131	128
$SSM + NL_{ordinary}$	1.084	28583	0.040	681	102	98

compute the first feasible solution when different reduction techniques were used. The test with normal sleep set method which compares and removes only time persistent pairs of transitions in combination with ordinary non-lazy runs was performed. This is represented as $SSM + NL_{ordinary}$ in the table 1. The combination of $SSM + NL_{ordinary}$ found the optimal solution. This is due to the fact that only time persistent transitions were pruned. No solutions could be found for the strict non-laziness setting when applied alone or in combination with the strict sleep-set method and the normal sleep-set method. It should be noted that the reduction techniques ordinary non-lazy runs and the normal sleep-set method are the only combinations which theoretically guaranty that the optimal solution is not pruned. The strict sleep set method in general should only be used as a heuristic.

6 CONCLUSION

A new and a straightforward approach to model scheduling problems in multi-product batch plants and to derive production schedules based on symbolic reachability analysis of TA has been introduced. The experimental results show that the method can solve scheduling problems in process industries efficiently within a limited computation time. The main aspect to be considered is the time required by the TA approach to specify and formulate the scheduling problem. It is relatively easy when compared to formulating it as MILP which are usually cumbersome. The efficiency of the approach to derive solutions with reduced computational effort have been achieved with different search space reduction techniques.

Future work on the TA based approach will focus on online scheduling problems in which the arrival time of jobs is not known in advance and subject to random perturbations.

ACKNOWLEDGEMENTS

The authors gratefully acknowledge the financial support by the Graduate School of Production Engineering and Logistics at the University of Dortmund.

REFERENCES

Abdeddaim, Y. and Maler, O. (2001). Job-shop scheduling using timed automata. *Computer Aided Verification (CAV), Springer*, pages 478–492.

Alur, R. and Dill, D. (1994). A theory of timed automata. *Theor. Comp.Science*, 126(2):183–235.

Barthomieu, B. and Diaz, M. (1991). Modeling and verification of time dependent systems using timed petri nets. *IEEE Transactions Software Engineering*, 126(2):259–273.

Bauer, N., Kowalewski, S., Sand, G., and Loehl, T. (2000). A case study: Multi product batch plant for the demonstration of scheduling and control problems. In *ADPM2000 Conference Proceedings-Hybrid Dynamic Systems*, pages 383–388. Shaker.

Behrmann, G., Fehnker, A., Hune, T., Peterson, P., Larsen, K., and Romjin, K. (2001). Efficient guiding towards cost-optimality in **UPPAAL**. In *Proceedings TACAS'01*, pages 174–188.

Godefroid, P. (1991). Using partial orders to improve automatic verification methods. *In Proc. 2nd Workshop on Computer Aided Verification*, 531 of LNCS:176–185.

Kallrath, J. (2002). Planning and scheduling in the process industry. *OR Spectrum*, 24:219–250.

Panek, S., Engell, S., and Lessner, C. (2005). Scheduling of a pipeless multi-product batch plant using mixed-integer programming combined with heuristics. In *European Symposium on Computer Aided Process Engineering, ESCAPE 15*, pages 1033–1038.

Panek, S., Engell, S., and Stursberg, O. (2006). Efficient synthesis of production schedules by optimization of timed automata. *Control Engineering Practice*, pages 149–156.

Panek, S., Engell, S., Subbiah, S., and Stursberg, O. (2007). Scheduling of multi-product batch plants based upon timed automata models. *Submitted to Comp. Chem. Eng*.

Pantelides, C. (1994). Unified frameworks for optimal process planning and scheduling. In *Proceedings 2nd conference Foundations of Computer Aided Process Operations*, pages 253–274.

A COMPARISON OF HUMAN AND MARKET-BASED ROBOT TASK PLANNERS

Guido Zarrella, Robert Gaimari and Bradley Goodman
The MITRE Corporation, 202 Burlington Road, Bedford MA 01803, USA
jzarrella@mitre.org, rgaimari@mitre.org, bgoodman@mitre.org

Keywords: Market-based multi-robot planning, intelligent control systems, distributed control, planning and scheduling, tight coordination, task deadlines, decision support.

Abstract: Urban search and rescue, reconnaissance, manufacturing, and team sports are all problem domains requiring multiple agents that are able to collaborate intelligently to achieve a team goal. In these domains task planning and assignment can be challenging to robots and humans alike. In this paper we introduce a market-based distributed task planning algorithm that has been adapted for heterogeneous, tightly coordinated robots in domains with time deadlines. We also report the results of our experiments comparing the robots' decisions with the decisions produced by ten teams of humans performing an identical search and rescue task. The outcome provides insight into the types of problems for which information technology can add value by providing decision support for human problem solvers.

1 INTRODUCTION

There are many modern problems that are not efficiently solved by a single human or robot. In domains like search and rescue, reconnaissance, and RoboCup, any attempt to solve the problem with a single robot may be inefficient, failure prone, or completely impossible. In these circumstances a team of agents must collaborate intelligently and task planning becomes central to the team success.

The extremes of multi-robot task planning and allocation algorithms are centralized and distributed approaches. In a centralized approach one agent plans the actions of the entire team and distributes the orders. In a distributed approach, each robot is responsible for creating its own plan using only local communication among robots. Centralized methods possess the key advantage of having all information needed to generate a globally optimal plan, while distributed approaches tend to be more scalable, robust to failure, and faster to respond to changes in the local environment. The ideal algorithm would combine features of both approaches to create a robust planning mechanism that is able to find a reasonable approximation of the optimal solution.

Past research into decentralized market-based task allocation protocols (Walsh et al., 1998; Dias, 2004; Lagoudakis et al., 2005) has been motivated by an attempt to design one such algorithm. In a market-based algorithm the robots bid against each other for tasks while acting rationally to maximize personal profit based on local calculations of cost and reward. This will move the entire team on average toward a globally efficient solution if the costs and revenues functions are properly constructed (Gerkey and Mataric, 2004). A market-based approach allows robot teams to reason efficiently about task allocation and resource management while preserving the ability of members of the team to adapt rapidly and robustly in the face of a dynamic environment. This technique mimics the flexibility of a free market economy by allowing ad-hoc teams to cooperate or compete opportunistically.

Prior research has demonstrated the effectiveness of variants of Dias' market-based TraderBots in several domains. The approach has been applied to tightly coordinated tasks that require heterogeneous, dynamically formed teams (Jones et al., 2006a). In this work two types of treasure hunting robots collaborate to simultaneously map an environment and detect the treasure within it. The TraderBots approach has also been used for task assignment in domains with time deadlines (Jones et al., 2006b), for example in homogeneous teams of fire-fighting robots completing tasks in which the reward for extinguishing a fire decays as a function of the elapsed time.

Yet another class of problems combines elements from the above domains. Collaborative

Time Sensitive Targeting (TST) is a domain requiring a diverse team of agents able to coordinate in discovering, assessing, prioritizing and solving new tasks within a very limited amount of time. This requires heterogeneous, dynamically formed teams that are both tightly coordinated and capable of reasoning about task deadlines. Search and rescue is one real-world example of a TST problem. For instance an avalanche rescue team's goal might be to "find each buried survivor and dig him or her out of the snow within sixty minutes." In this case searchers and diggers need to form dynamically changing and complementary teams to rescue as many survivors as possible within a limited time.

Time Sensitive Targeting can be a difficult problem solving task for humans as well as robots. Frequently decisions must be made about how to re-evaluate team strategy to make the best use of scarce resources. This makes TST an ideal testbed for a market-based task planning and allocation algorithm.

This paper describes our attempt to design and evaluate the first market-based planning system capable of reasoning in situations requiring tightly coordinated, deadline aware agents. In Section 2 we describe the specifics of our simulated Time Sensitive Targeting domain. We introduce our planning algorithm in Section 3. In Section 4 we discuss our experiments involving teams of humans attempting to solve a TST problem. In Section 5 we contrast the human and robot results, and in Section 6 we present our conclusions about the potential for the application of information technology to benefit teams of human decision makers.

2 TST SCENARIO

The central element of solving a Time Sensitive Targeting problem is the ability to assess and respond to emerging tasks within a limited window of time. The typical TST task requires a coordinated effort between a large number of specialized information gathering and action taking agents. Furthermore it is essential that the team is able to continually reprioritize its goals as new information arrives from the noisy and rapidly changing environment.

We designed a simulated TST scenario to use in our task planning and problem solving experiments. Our scenario is a type of Search and Rescue problem in which agents attempt to locate, investigate, and rescue six simultaneously moving targets before each target's time deadline expires.

The premise of the scenario is that the Coast Guard is responsible for monitoring three areas of ocean for sick or injured animals. The Coast Guard is provided with a fleet of specialized vehicles such as helicopters, boats, and submarines. The goal is to use these vehicles to find, diagnose, and rescue a series of endangered animals. In our experiments the fleet of vehicles was controlled either by a small team of humans or by our market-based robot task planning algorithm.

Over the course of the 90 minutes of an exercise, the Coast Guard receives messages containing reports of the general locations where distressed animals have been sighted. A message provides the type of animal, an approximate latitude-longitude, a time deadline for task completion (e.g. cure the sick manatee within 30 minutes or it will die), and the relative value of the task (represented by the maximum reward offered for task completion).

The Coast Guard's vehicle fleet includes a heterogeneous collection of robots. There are three main categories of vehicles.

Radar Sensors are planes and boats equipped with radar or sonar sensing capabilities. They are generally very fast and have large sensing range, so they can get to a location quickly, pinpoint where an animal is located, and track an animal as it moves. They can share the information they gather with other teammates. Due to the limitations of radar, this type of sensor is not able to determine an animal's species or diagnose an illness.

Video Sensors include boats and helicopters with visual sensing capabilities. They are able to identify animal types and diagnose diseases. They can also report the information they have gathered to the rest of the team. However they tend to move slowly and have limited sensing range, so they are best used in tandem with other sensors.

Rescue Workers are boats or submarines outfitted with equipment for capturing or curing an animal in distress. This is the only type of vehicle capable of saving an animal once it has been located. They are generally about as fast as radar sensors, but they have no sensors of their own. They must rely on reports from the sensor robots for navigation data. Also, they are only allowed to assist an animal after the proper diagnosis has been made by a video sensor.

The Coast Guard has multiple robots in each group. Even within groups there are variations of individual characteristics such as speed or sensor range. There are 33 vehicles in total, divided between three separate areas of ocean.

Because of the specialization of the robots, they are required to form ad hoc teams to fully complete any task. Each team must, at a minimum, consist of two robots: a video sensor to find the animal and make the diagnosis, and a rescue worker to assist the animal. A radar sensor is not required but its speed

and sensor range can greatly reduce the overall time needed for a team to assist an animal.

3 MARKET-BASED ALGORITHM

We chose to develop our market-based multi-robot task planning algorithm within a controlled simulation environment. The entire package was written in Java using the JADE agent framework (Bellifemine et al., 2001). Our agents used only local robot-to-robot communication to implement the task planning protocol. The planning algorithm shares many similarities with TraderBots and other existing market-based approaches. Our goal was to extend the existing approaches to be capable of performing planning in domains with both task deadlines and tightly coordinated ad hoc teams.

The agents in our simulation trade labor for a fictional currency. An agent earns revenue for the successful completion of one of the tasks the team has been asked to perform, but only if the task is completed before the time deadline arrives. The agent incurs costs in the process of doing work to achieve a goal; these costs are proportional to the amount of time spent working towards the task. An agent also condsiders the opportunity cost (Schneider et al., 2005) of agreeing to perform a task. The self interested agents will only bid on a task if the potential revenue outweighs the sum of the impending costs. Agents buy and sell tasks from each other, forming efficient, specialized teams in the process. The cost and revenue functions we have chosen are conducive to fostering teams that solve problems as quickly as possible without over-committing the existing resources.

This section of the paper contains a high-level description of our implementation. (Gaimari et al., 2007) provides more detail about the algorithm.

3.1 Agents

The *TraderAgent* is the building block of the robot economy. Each TraderAgent controls one robot in the simulation environment. A TraderAgent's primary job, as the name implies, is to trade tasks.

Any agent that owns a task may put it up for auction, announcing the maximum reward it is prepared to pay. Other TraderAgents that wish to bid for the job may do so, and after one round of bidding the seller announces the winner. In standard re-auctioning this passes ownership of the task to the buying agent; this agent must have a robot with the same capabilities as the seller. The selling agent's only responsibility thereafter is to pay the promised bid to the buyer upon completion of the task.

In our system there is an additional type of re-auctioning that occurs. Since the robots must work together in teams to complete the tasks, some re-auctions are for the purpose of teambuilding among agents with different capabilities. In this case both agents retain ownership of the task. For each task a TraderAgent owns, there is a corresponding list of the teammates it is working with. If an agent owns multiple tasks it can belong to multiple teams.

New tasks are given to a special agent that executes the initial auction. This agent does not control a robot in the simulation.

3.2 Bidding on Auctions

When a TraderAgent receives an auction announcement, it performs the following steps:

- It calculates the estimated cost for performing the task. In our scenario the cost is given by the amount of time required to accomplish a task. Since the description of a task provides noisy and imprecise information about the location of an animal, costs cannot be determined exactly in advance. The agent prepares a cost estimate based on its technical abilities and current location.
- It calculates the opportunity cost associated with accepting responsibility for the task. This represents the likelihood that the agent will be able to win hypothetical future tasks. Robots with especially unique abilities will have higher opportunity costs than more common types of robots. Opportunity cost is also affected by the robot's location on the map, as some areas are more desirable for finding work than others.
- It calculates the desired profit margin. This is a function of the opportunity cost and the difference between the offered reward and the estimated cost. Robots with low opportunity costs will lower their desired profit margin in an attempt to increase the chance of winning the current auction.
- It calculates the final bid amount and places a bid if the cost plus the desired profit margin is less than the reward offered by the seller.

3.3 Collecting Payment

Once a task is completed, each TraderAgent reports that fact to the agent it bought the task from, asking to be paid. Domains with tightly coordinated, heterogeneous teams and time deadlines require special handling of payment allocation. In this case the teams are made up of robots that do their jobs at

greatly different speeds. Slower robots can lead to much higher costs and lower rewards than a faster robot may have originally estimated. If the cost estimations are too inaccurate, the ability of agents to prioritize different tasks is damaged.

In our system an agent penalizes its teammates when the team underperforms expectations. Each agent requests the amount of payment agreed upon during the bidding process. As the payments are distributed, each agent compares its actual cost to the estimated cost it had initially planned upon. The difference between these is deducted from the amount paid to the next agent. This agent then adds the difference in its own actual cost and original estimate, plus the amount it was penalized by its seller. The penalty moves down the chain in this fashion until it finally ends where it belongs, on the slowest member of the team. These payments reflect the amount of money the original agents would have bid had they had known the true cost of working with slower robots. This penalty system provides feedback that allows the robots to learn improvements to their cost estimation and bidding practices.

4 HUMAN EXPERIMENTS

We tested the performance of teams of people on an isomorphic version of the Coast Guard search and rescue problem. The performance results of these teams of humans are directly comparable to the performance results of our market-based robot teams.

In these experiments, each team consisted of three college educated adults. The teams were mixed sex and made up of computer literate participants between the ages of 28 and 65. The members of the teams were provided with computer tools allowing them to view maps of the environment and control the movements and actions of the simulated robot vehicles. The participants were working in the same room and were permitted to speak with each other but were not allowed to look at the others' computer displays. Each member of the team was randomly assigned a unique and complementary role.

The **Intel Officer** acted as the team leader and was responsible for coordinating the team response to targets assigned to the group. This officer received the messages containing the rumored locations of new targets. The messages also specified a time deadline by which the task had to be completed. The intel officer was then expected to share this new information with the team and monitor the group's progress toward the goal.

The **Sensor Analyst** commanded the fleet of 20 heterogeneous sensor devices, including video equipped helicopters and radar planes. The sensor analyst was responsible for choosing which sensors to use, for ordering changes in sensor paths, and for monitoring the state of each sensor to check for newly detected items.

The **Rescue Worker** commanded a fleet of 13 heterogeneous rescue vehicles. This analyst was responsible for choosing which rescue vehicles to deploy, for ordering changes to each vehicle's path, and for giving the official order to rescue an animal.

As in the robot experiments, the teams were expected to locate and positively identify each target using their sensors before rescuing the animal. The experiment was an exercise in communication and team problem solving. Successful prosecution of a target was dependent on the participants' ability to 1) share relevant information without distracting each other from the task at hand, 2) interpret the state of the environment in a timely fashion, and 3) choose appropriate actions to execute. The simulation was developed as a simplification of real world exercises performed by similar teams of TST analysts (Goodman et al., 2005).

5 EXPERIMENTAL RESULTS

We evaluated the performance of the human and robot teams on our search and rescue TST scenario. Ten teams of three people attempted the problem. Each experiment lasted for 90 minutes. During this time, six targets were assigned to each team. The first three targets were assigned at 15 minute intervals, and the last three targets were assigned at 5 minute intervals. Each target had a time deadline between the 80^{th} and 90^{th} minutes of the experiment.

Table 1 shows the number of tasks completed by each human team. The best teams completed four of the six tasks before the time deadline. The worst teams were unable to successfully complete any of the tasks. The average number of tasks completed by the ten teams was 1.9, and the median was 2. In all, the teams of humans completed 32% of the tasks.

Table 1: The number of tasks, out of 6, completed by each group before the time deadline.

Team #	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
# Tasks Finished on Time	2	3	4	1	3	0	0	0	4	2

Table 2 shows the number of teams that were able to successfully complete each task before the time deadline. Note that two of the tasks (#4, #6)

were not completed by any of the teams. Another task (#2) was completed by all teams except for those groups that did not complete any tasks. These figures indicate that in general the tasks were not trivial to solve by teams of humans attempting the assignment, and that there was a good mix of difficulty levels in the problems presented to the teams.

Table 2: The number of teams, out of 10, that completed each task before the deadline.

Task #	1	2	3	4	5	6
# of Successful Teams	4	7	3	0	5	0

Table 3: The tasks completed by the autonomous robots before the deadline.

Task #	1	2	3	4	5	6
Solved before Deadline?	Y	Y	Y	Y	Y	N

The results of the robot team are displayed in Table 3. The team of robots completed five of the six tasks, a success rate better than best of the human teams. This demonstrates the ability of the robots to apply effective team building and task assignment strategies. We also use length of time before solution to compare robot team performance to human team performance, shown in Figure 1.

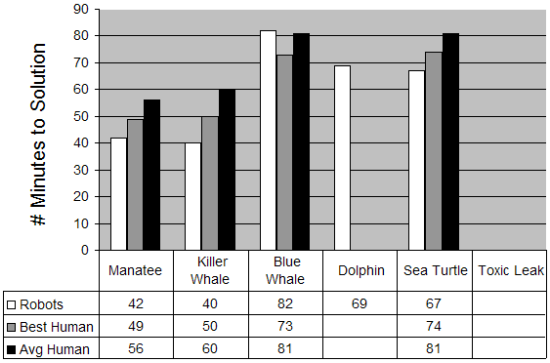


Figure 1: A comparison of the time to solution for the robot team, average human team, and best human team. Lower is better.

The robot teams compare very favorably to the human teams. The simulated agents were much faster than the best human teams in three of the four tasks that were solved by both humans and robots. The agents also were able to complete the Dolphin task, which none of the ten human teams had successfully accomplished within the time deadline. The simulated agents did fail to complete one task, but none of the human teams were able to successfully complete that task either.

6 APPLICATIONS AND CONCLUSION

The ad hoc teams of distributed market-based task planners demonstrated better performance on a simplified Time Sensitive Targeting task than the teams of humans attempting the same task. This result demonstrates that it is feasible to use our planning algorithms on tightly coordinated and time constrained tasks. The result is especially interesting in light of the fact that real-life problem solvers, such as military TST analysts, are humans collaborating in ad-hoc teams to attempt to combine forces into one integrated, efficient system.

What are the reasons for these differences in performance, and how can we use advances in information technology to improve human efficiency? Humans have an advantage over computers in that a lifetime of interactions with other humans allow them to plan and coordinate actions without the need for a formal communication and negotiation structure. Humans are also naturally able to integrate new information into the planning process in an online manner. Therefore the results described above are at least moderately surprising.

However, one key limiting factor on human performance is that humans have limited attention resources. It isn't possible for a single person to attend to the output of all twenty sensors simultaneously. As the number of concurrent tasks increase, human teams can suffer from increased cognitive load, which can dramatically affect a team's ability to respond to new information in a timely manner. One example of this can be seen in our human TST experiments, in which the average time delay between receiving and reading a new e-mail message increased steadily as more concurrent tasks were added.

In essence, the teams of humans are exhibiting the same drawbacks of a centralized multi-robot planning algorithm. Information from sensors must propagate to the top of the chain of command before a plan can be implemented that reflects changes in the state of the task. For some domains this is an adequate solution; unfortunately humans do not "scale" well to larger scenarios in which attention resources must be divided between larger numbers of targets. The results of our experiments demonstrate that TST teams can struggle when forced to make decisions about which targets are most worth pursuing given limited attention and

resources. Real world teams are routinely forced into this situation. At SIMEX, a realistic TST simulation that uses real analysts from various government forces, 145 vehicles are manned by 30 operators pursuing any number of targets (Loren, 2004).

The market-based robot planning system, in these situations, is able to benefit from its distributed nature. As each autonomous agent receives updates on the state of the environment, this information is immediately propagated to the affected agents. This means that new tasks or newly sensed targets are promptly incorporated into the team plan. In the robot teams, the performance bottleneck is the quality of the decision making process rather than the availability of relevant data.

It is unreasonable to suggest that intelligent agents can replace the human decision makers in high risk Time Sensitive Targeting environments. The results from our simplified and noise-free environment can't necessarily be extrapolated to apply in far more complex real-world situations. The research does however indicate that there is value in applying intelligent control systems and other information technology to complement human decision makers by mitigating human weaknesses.

Our future work in this domain is focused on incorporating the task planning agents into an intelligent cognitive aide. The aide will draw attention to relevant events and changes in the environmental state. We could also use this cognitive aide to improve training methods by teaching decision makers to focus their attention on the most critical plan-changing events.

We have shown it is possible to use intelligent control systems to improve upon the results exhibited by teams of human decision makers. Our hope for the future is that it is possible to combine human and robotic planning methods to yield even better results.

ACKNOWLEDGEMENTS

The MITRE Technology Program supported the research described here. We are also grateful for the assistance of Brian C. Williams and Lars Blackmore at the Massachusetts Institute of Technology.

REFERENCES

Bellifemine F., Poggi A., Rimassa G. (2001). Developing multi-agent systems with a FIPA-compliant agent framework. *Software-Practice and Experience*, *31*, 103–128.

Dias M.B. (2004). TraderBots: A New Paradigm for Robust and Efficient Multirobot Coordination in Dynamic Environments. Doctoral dissertation, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA.

Gaimari, R., Zarrella, G., Goodman, B. (2007). Multi-Robot Task Allocation with Tightly Coordinated Tasks and Deadlines using Market-Based Methods. Proceedings of *Workshop on Multi-Agent Robotic Systems (MARS)*.

Gerkey B. and Mataric M. (2003). Multi-robot Task Allocation: Analyzing the Complexity and Optimality of Key Architectures. Proceedings of *IEEE Conference on Robotics and Automation*.

Goodman B., Linton F., Gaimari R., Hitzeman J., Ross H., Zarrella G. (2005). Using Dialogue Features to Predict Trouble During Collaborative Learning. *User Modeling and User-Adapted Interaction*, *15* (1-2), 85-134.

Jones E., Browning B., Dias M. B., Argall B., Veloso M., Stentz A. (2006a). Dynamically Formed Heterogeneous Robot Teams Performing Tightly-Coordinated Tasks. Proceedings of *International Conference on Robotics and Automation*.

Jones E., Dias M. B., Stentz A. (2006b). Learning-enhanced Market-based Task Allocation for Disaster Response. Tech report CMU-RI-TR-06-48, Robotics Institute, Carnegie Mellon University.

Lagoudakis M., Markakis E., Kempe D., Keskinocak P., Kleywegt A., Koenig S., Tovey C., Meyerson A., Jain S. (2005). Auction-Based Multi-Robot Routing. *Robotics: Science and Systems*. Retrieved at http://www.roboticsproceedings.org/rss01/p45.pdf

Loren, Lew. (2004, Fall). Experimentation and Prototyping Laboratories Forge Military Process and Product Improvements. *Edge Magazine*. Retrieved at www.mitre.org/news/the_edge/fall_04/loren.html

Schneider J., Apfelbaum D., Bagnell D., Simmons R. (2005). Learning Opportunity Costs in Multi-Robot Market Based Planners. Proceedings of *IEEE Conference on Robotics and Automation*.

Walsh W., Wellman M., Wurman P., MacKie-Mason J. (1998). Some Economics of Market-Based Distributed Scheduling. Proceedings of *International Conference on Distributed Computing Systems*.

A HYBRID INTELLIGENT MULTI-AGENT METHOD FOR MONITORING AND FAULTS DIAGNOSIS

Gang Yao and Tianhao Tang
Department of Electrical & Control Engineering, Shanghai Maritime University
1550 Pudong Road, Shanghai, 200135, P. R. China
thtang@cen.shmtu.edu.cn, shakesteel@hotmail.com

Keywords: Multi-agent system, Monitoring and diagnosis system, Data mining, Fuzzy neural networks.

Abstract: This paper presents a hybrid intelligent multi-agent method for monitoring and faults diagnosis. A new diagnosis process, combined with data mining and neural networks, are discussed as well as the functions and structure of agent which implements these algorithms. At last, some simulation results are shown to demonstrate the efficiency of the proposed system.

1 INTRODUCTION

The rapid development of modern industry calls for safer and more efficient control processes. Monitoring and faults diagnosis systems, specifically combined with artificial intelligent technologies, are implemented for state monitoring, trend predicting and fault diagnosis. Thereby, it is possible to improve the system efficiency and to guarantee the operation safety in the control process (Edgar 2000).

A general overview about distributed artificial intelligence in industry was given in (Parunak 1994). This paper reviewed the industrial needs for distributed artificial intelligence, and gave special attention to the needs arising from systems for manufacturing, scheduling and control. Since then more and more researches and contributions have been done in this field.

However, the complexity of the monitoring and diagnosis system is growing with the increasing complexity of industrial plants. To keep the monitoring and diagnosis system effective, it is essential to encapsulate different tasks and to define strict interfaces between plant components and between components of the monitoring and diagnosis system, although it is quite difficult. To guarantee flexibility -- changing needs in case of an industrial application, the monitoring and diagnosis system has to be configurable and expandable without the need of modifying any line of code (Luder 2001). The diagnostic knowledge about an industrial process is available on different parties (process specialists, component manufacturers, etc.). A modern monitoring and diagnostic system should be able to integrate the diagnostic knowledge from all available sources, even if different diagnostic mechanisms are applied. To achieve an overall diagnosis of a control process, several diagnostic tasks have to be performed in parallel. This requires new strategies to handle diagnostic conflicts that might occur between different diagnostic results.

Multi-agent system (MAS), about which rapid progress has been made, is an important research branch in distributed artificial intelligence (DAI) parallelized with distributed problem solving (DPS). Possessing modularity, adaptability and other attractive characteristics, MAS drew much attention in recent years and is adopted by many researches in monitoring and diagnosis system.

This paper presents a hybrid intelligent mutli-agent method for monitoring and faults diagnosis, which separates fault diagnosis process into several steps excuted by different types of agents. The macroscopical architecture of the MAS system and microcosmic structure of an agent are designed in section 2. Then, the intelligent monitoring and fault diagnosis process is presented in section 3. At last, the simulation experiment, applying the proposed method in marine engine room, is carried out in section 4.

2 SYSTEM ARCHITECTURE

2.1 Architecture of Proposed System

The framework of hybrid intelligent multi-agent method, with hierarchical and federal organized software agents that are responsible for different tasks, is presented in figure 1.

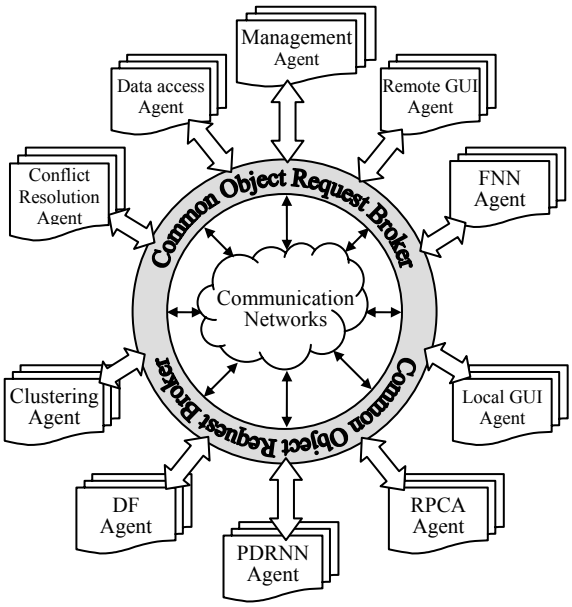


Figure 1: Architecture of MAS-based monitoring and diagnosis system.

As shown in figure 1, many agents with different capabilities are connected together by accessing common object request broker (CORBA) functionalities and through communication networks to form a multi-agent society. In this society, each individual has a special 'survival skill' that can work autonomously and independently. For example, when this MAS is connected to a control system, the data access agent, who has the skill of accessing database, it can get signals periodically and automatically from given sensors and store them into a record set with the intelligence of maintenance the integrity of database.

On the other hand, like human beings, agents in the figure trend to seek cooperation to fulfil more difficult task if they believe that better rewards will gain by cooperation or the job assigned to the agents is impossible to achieve with their own capability. A cooperation coalition will form successfully if following precondition is met

$$v(A_1)+ v(A_2) \le v(A_1+A_2), (A_1 \cap A_2=\phi) \quad (1)$$

Where $v(A_1)$ is the benefit gained by coalition A_1 after A_1 accomplished a job.

Management agent is the one who take charge of negotiation within the formation of a cooperation coalition. When a task is received, management agent decomposes the task into sub-jobs or steps if necessary, and then adopts contract net protocol to distribute them to appropriate agents to form a cooperation coalition. Other agents decide whether to respond to the bidding or not according to the job been doing, priority and rewards. After the mission is accomplished, the cooperation coalition will dismiss automatically.

This MAS approach will bring us following advantages: Modularity and scalability, instead of adding new capabilities to a system, agents can be added and deleted without breaking or interrupting the process; Adaptability, agents have the ability to reconfigure themselves to accommodate new changes and faults; Concurrency, agents are capable of reasoning and performing tasks in parallel, which in turn provides more flexibility and speeds up computation; Dynamics, agents can dynamically collaborate to share their resources and solve problems and finally, Reliability, MAS are more fault-tolerant and robust than traditional AI systems.

2.2 Architecture of an Intelligent Agent

At agent level, all the agents in proposed method have the same hybrid behaviour architecture, where the agents are capable of reactive and deliberative behaviours. In general, the agents should be neither totally deliberative nor totally reactive. If they are only reactive, they cannot reason about their actions and will not be able to achieve any sophisticated behaviour; if they are just deliberative they may never be able to act in time. The proposed architecture is based on horizontal layering where all layers are connected to the perception and actuation of the agents with the environment. Figure 2 shows the proposed agent architecture in monitoring and diagnosis system.

In figure 2, an agent can collect information from two channels: the perception module, which apperceives from the ambient and to check the influence of last action, and the communication module, which receives message from other agents. All the perception information is distinguished as 'urgent' or 'not urgent' based on the signal type, priority, security policy and experience in order to trigger corresponding response mechanism.

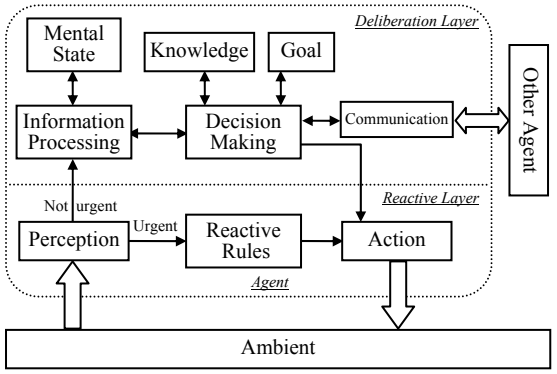


Figure 2: Architecture of an agent.

If an urgent message is received, the reactive layer will be triggered, and the agent will execute according to the most similar rule in rules library without thinking. The reactive rules library could be modified in accordance with the experience automatically.

If the message is not urgent, the agent will 'think' for a while about how to respond. In this period, agent uses its special ability to process this information and then make decision with the consideration of mental state, knowledge and its goal. After the agent's action is executed, if the action really works, the agent will record this action as a paradigm into reactive rules library and update the mental state, knowledge base if necessary.

When the agent finds the job got from the message is too difficult to accomplish, three options are available: (1) if the agent know who can help it, it will ask for help directly to that agent; (2) if the agent has no idea who is the right agent, it will contact the management agent to try to organize a cooperation coalition; (3) if no one responds its request, abandoning the goal is its last choice.

The special capability mentioned above is the agent's 'survival skill' encapsulated in information processing module (IPM). Different method in IPM determines different type of agent. As shown in figure 1, ten kinds of agent are designed in this system:

(1) Local and remote GUI agent: local and remote graphical user interfaces (GUI) are used by the operator users to display monitoring and diagnosis results, initiate diagnostic processes, give a phonic or flaring alarm, and receive user's instructions locally and extendedly.

(2) Management agent: management agent is used to decompose task and start organizing cooperation as mentioned in section 2.1.

(3) Conflict resolution agent: a conflict resolution mechanism is required to investigate whether the diagnostic results, as reported by different diagnostic agents, are contradicting or completing each other. The diagnostic agents do not communicate with each other to merge their knowledge, but do report their diagnosis to a conflict resolution agent. For this purpose, the history credit evaluation of a diagnosis agent is important. Beyond this, knowledge of relations among the components and among the possible failures which may be related within the components, need to be well known (H.Worn 2002).

(4) Directory facilitator agent: the directory facilitator (DF) agent is responsible for communication and agent management. It can provide the naming service, represent the authority in the platform and also provide Yellow Pages service by means of which an agent can find other agents providing the services he requires in order to achieve his goals. All the capabilities of the registered monitoring and diagnostic agents and the available CORBA functionalities are managed by the facilitator agents.

(5) Data access agent: what data access agent can do has discussed as an example in section 1.

(6) Clustering agent, Relative Principal Component Analysis (RPCA) agent, Parallel Diagonal Recurrent Neuron Network (PDRNN) agent and Fuzzy Neural Network (FNN) agent: these agents are dealing with monitoring and diagnosis process which will be discussed in next section.

3 INTELLIGENT MONITORING AND DIAGNOSIS PROCESS

Faults diagnosis for complex control system is the process of mining valuable omen variables from mass data collected by sensors and mapping omen variables to faults modes. Thereby data mining plays an important role in diagnosis. In this paper, a new hybrid intelligent monitoring and diagnosis method is proposed in figure 3. This method divided the process of data mining and fault mode mapping into several independently data fusion modules, which are implemented by agents:

(1) Database: database is made up with two main storage areas, which correspond to history and online data access respectively. History data are used for intelligent data mining, executed by collaborated agents, and real time data are collected by the data access agent from sensors.

(2) Pre-processing module based on clustering methods: This module is executed by clustering agent. Data selection and data mining usually do not need all the data, some data object and propriety has no contribution to the modelling, on the contrast they will greatly reduce the efficiency of data mining, even will lead to the variation of data mining result. In this case, it is very necessary to select data effectively. Pre-processing module based on clustering methods can select the preventative points as feature data and filter some fake data.

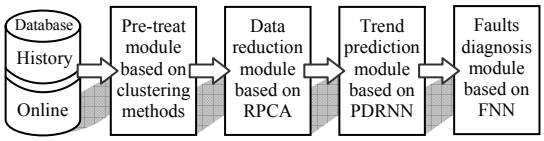


Figure 3: Data mining based intelligent information fusion method.

(3) Data reduction module based on RPCA: Data reduction will figure out the essential data set to shrink the scope, which is the further step of reduction based on data selection. Data reduction also is called as data enriching, the process of which is transferring the original data set to the more compact data set without losing any semantics. Herein Relative Principal Component Analysis Algorithm (RPCA) is adopted and encapsulated in the IPM of RPCA agent, which can avoid the parameters having bigger absolute values or variation values to play the great role when getting main element, which is the shortage of PCA method. For RPCA, more representative main element can be got from the data array made up with evenly distributed system variables series. This module is majorly used to reduce data dimension and extract system feature (Tianhao Tang 2005).

(4) Trend prediction module based on neural network: Prediction mode can predict some phenomena or data value. Parallel Diagonal Recurrent Neuron Network (PDRNN), which is implemented in PDRNN agent, can realize multiple dimension and parallel time frequency prediction, and it has the high prediction precision to make its result good for the fault trend analysis. The architecture of this neural network is shown in figure 4. The transfer function of neuron adopts Sigmoid, and training method is back propagation algorithm. Details about this neural network refer to (Tianzhen Wang 2004).

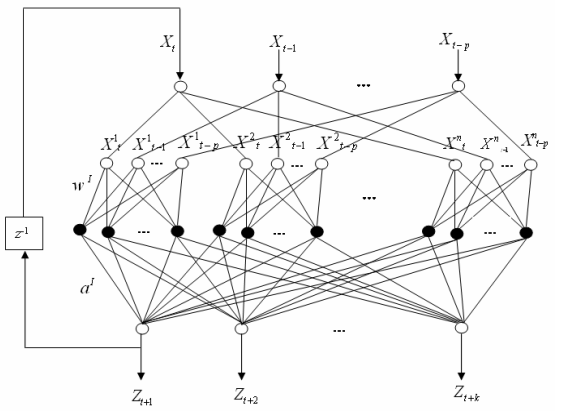


Figure 4: Architecture of PDRNN.

(5) The last step is to do system fault detection based on omen variable and parameters prediction. The core issue of fault diagnosis is to abstract signal feature and to establish the relationship between the features and different functional states. More complex modern industry system has more complex faults. One fault often demonstrates many omens, while one omen maybe includes information about many kinds of faults. The relationship between omen and fault is hard to be expressed by precise mathematic expression, so fuzzy logic diagnosis method can be used. The basic principle of fuzzy logic diagnosis is to establish the fuzzy relation matrix $\widetilde{R}$ between fault mode (cause) and omen variable according to the known information, then to select omen membership vector X. The fault mode membership matrix Y can be calculated from fuzzy relation equation Y=X* $\widetilde{R}$, thus fault causes can be diagnosed.

The uncertain relationship between fault and omen can be well expressed in fuzzy diagnosis, but the relation matrix $\widetilde{R}$ is hard to be established. On the other hand, fault diagnosis can be regarded as a kind of pattern recognition which maps the fault omen to fault causes. The relation matrix $\widetilde{R}$ reflects the mapping. For the complex system, the mapping is non-linear. A fuzzy neural network (FNN) is used to establish the matrix $\widetilde{R}$, as shown in figure 5

This FNN is realized in FNN agent, and two new fuzzy operators, the generalized probability sum operator and the generalized probability product operator, are used as transfer function to express the concepts of the generalized union and the generalized intersection calculating. Details about the FNN refer to (Tang, etc., 1998).

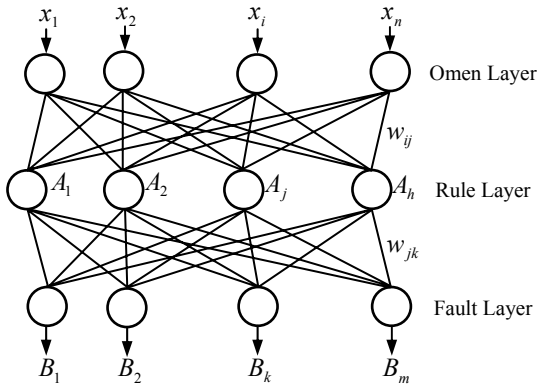


Figure 5: Architecture of fault diagnosis neural network.

4 SIMULATION EXPERIMENT

The hybrid intelligent multi-agent monitoring and diagnosis system mentioned above has been used in marine engine room, which is a typical complex and distributed system.

Marine engine room is made up principally with main engine remote control & protection system (MECS), auxiliary engine control system (AECS), ship power station automatic control system (PSCS), boiler automatic control system (BCS), cabin monitoring and alarm system (CMAS), and pump control system (PCS), etc. For every sub systems, a cooperative coalition is formed to implement monitor and diagnosis.

The overall state monitoring and fault diagnosis system for marine engine room is shown in figure 6. The architecture of a cooperative coalition is shown in figure 7.

As mentioned before, there are several different types of agents sharing the same architecture but having different capability in the proposed system, which means that the connotation of IPM is different according to the functionality of the agent. Thereby, in figure 6, the architecture of all kinds of agents is implemented in Jade, but the inner methods of IFM are coded in different programming language with consideration of coding convenience.

The monitor and alarm client interface, running on local and extended GUI agent, and the fuzzy neural network in FNN agent are coded in Visual Basic 6.0 (VB6). Other algorithms involving in the process of intelligent data mining are coded in Matlab 6.5.

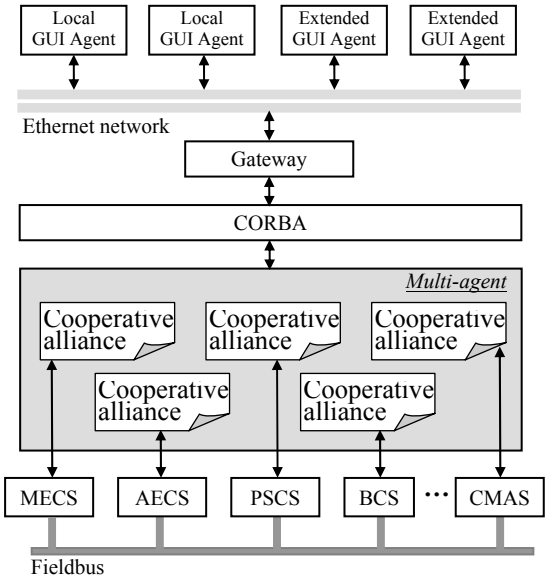


Figure 6: Architecture of marine engine room state monitoring and fault diagnosis system.

JMatLink, a small toolkit to connect Java with Matlab, is used to call for the data mining function in an m-file for Java monitoring agent.

To implement calling VB methods and forms from Java, Java Native Interface (JNI) mechanism is utilized.

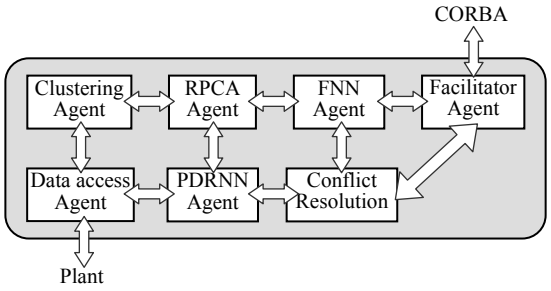


Figure 7: Architecture of a cooperative coalition.

In this experiment, several faults of main engine were adopted to test the faults diagnosis system. Ten parameters of main engine, such as engine cylinder exhaust gas temperature, maximum cylinder pressure, etc, are chosen as omen variables.

Table 1 shows the comparison between samples of normal state and three faults state and diagnosis results. The acronyms of omen variables are interpreted in Table 2.

Table 1: Fault diagnosis results.

	Normal state		Main oil pump abrasion	
	Sample	Result	Sample	Result
Pmax	0.50	0.485	0.05	0.052
Pcs	0.50	0.488	0.35	0.405
Tcs	0.50	0.499	0.35	0.381
Tex	0.50	0.446	0.20	0.228
Ntk1	0.50	0.486	0.35	0.408
Ntk2	0.50	0.496	0.35	0.408
Pti	0.50	0.495	0.35	0.416
Tto	0.50	0.513	0.35	0.404
Tko1	0.50	0.497	0.35	0.386
Tko2	0.50	0.497	0.35	0.396

Table 1: Fault diagnosis results (Cont.).

	Fuel injection late		Fuel injection pipe leak	
	Sample	Result	Sample	Result
Pmax	0.05	0.063	0.05	0.051
Pcs	0.80	0.715	0.20	0.232
Tcs	0.65	0.534	0.35	0.378
Tex	0.95	0.931	0.05	0.141
Ntk1	0.65	0.613	0.35	0.379
Ntk2	0.65	0.612	0.35	0.378
Pti	0.65	0.604	0.35	0.411
Tto	0.65	0.591	0.50	0.512
Tko1	0.65	0.596	0.35	0.407
Tko2	0.65	0.595	0.35	0.402

Table 2: Interpretation of omen variables.

Pmax	maximum cylinder pressure
Pcs	scavenging air pressure
Tcs	scavenging air temperature
Tex	cylinder exhaust temperature
Ntk1	rpm of No.1 turbocharger
Ntk2	rpm of No.2 turbocharger
Pti	inlet pressure of turbocharger
Tto	outlet temperature of turbocharger
Tko1	No.1 air cooler outlet temperature
Tko2	No.2 air cooler outlet temperature

From the table, we can see that diagnosis agent can get close results compared with the samples, and it means that fault could be detected exactly. Besides these faults in Table 1, dealing with other faults, diagnostic agent can get similar results.

5 CONCLUSIONS

This paper describes a concept of building a hybrid intelligent monitoring and diagnosing system for complex control process based on the application of MAS, and also proposed a new fault diagnosis process integrates several algorithms implemented within the MAS method, which allows the flexibility, the extendibility, and a cost-effective development of the system. Details about the overall architecture, algorithm encapsulated in IPM, and coding tools are discussed. And at last, some simulation experiment results are given to demonstrate the efficiency of the presented system.

ACKNOWLEDGEMENTS

This work was supported by National Natural Science Foundation of China (60572051) and the project from Shanghai Education Foundation (05FZ04).

REFERENCES

Edgar, T. F., Dixon, D. A., and REKLAITIS, G. V., 2000, Vision 2020: Computational Needs of the Chemical Industry, (University of Texas Press).

Parunak, V., 1994, Applications of distributed artificial intelligence in industry. In O'Hare and Jennings, (eds) Foundations of Distributed Artificial Intelligence (Chichester: Wiley Inter-Science).

Luder, A., et al., 2001, Industrial requirement and overall specification. Prepared within the PABADIS IST research project no. IST-1999-60016. Available at www.pabadis.org.

H.Worn,et al, 2002, A distributed multi-agent architecture for monitoring and diagnosis

Tianhao Tang, Tianzhen Wang. 2005, ANN-based multiple dimension predictor for ship route prediction. Proceedings of ICINCO 2005, Barcelona, Spain

Tianzhen Wang, Tianhao Tang. 2004 A Mult-dimension Predictor based on PDRNN . ICARCV, P1359-1364

Tang, T. et al 1998. A fuzzy and neural network integrated intelligence approach for fault diagnosing and monitoring. Proceedings of the 1998 UKACC International Conference on Control, vol.2, pp.975-980. Swansea, UK.

Tianhao Tang, Yao Gang. June 27-29, 2005, A Fault-tolerant Control Method Based on Adaptive Fuzzy Neural Networks for Ship Control System, 2005 International Conference on Control and Automation, Budapest, Hungary

ENHANCING KAPPA NUMBER CONTROL IN DOWNFLOW LO-SOLIDS^{TM} DIGESTER USING DIAGNOSIS AND MODELLING

Timo Ahvenlampi
University of Oulu, Systems Engineering Laboratory, 90014 University of Oulu, Finland
timo.ahvenlampi@oulu.fi

Rami Rantanen
Metso Automation Inc., Process Automation Systems, Tampere, Finland
rami.rantanen@metso.com

Keywords: Fault tolerant, pulp industry, monitoring, controllability.

Abstract: In this study, Kappa number prediction and diagnosis in continuous Downflow Lo-Solids^{TM} cooking application is investigated. Gustafson's Kappa number model is applied for the prediction of the blowline Kappa number. New cooking temperature set point is solved iteratively based on the difference between the predicted and target blow-line kappa numbers. The main active variables for the Kappa number are monitored using self-organizing map (SOM). The diagnosis and Kappa control are combined into a fault tolerant system. The data is collected from industrial continuous Downflow Lo-Solids^{TM} cooking digester. Good results were achieved using the proposed approach.

1 INTRODUCTION

The question how the overall system of the sub processes and chains of sub processes can be improved, by means of fault diagnosis (see e.g. (Isermann, 1997), (Venkatasubramanian et al., 2003a) and (Venkatasubramanian et al., 2003b)), has not fully answered. In particular, this holds for demanding process conditions such as found in chemical and mechanical pulping. Modern processes generate a lot of information, which can be used for improving the operation of the process and quality of the products. This can be accomplished by combining expert knowledge, modelling, control and fault diagnosis.

The pulp digester is very important unit operation in the chemical pulping plant. The control actions in the digester have effects to the entire fiber line operations. Also the quality of the pulp in the digester should be achieved with minimal cooking costs. (Leiviskä, 2000) The quality of the chemical pulping is characterized e.g. by the pulp's strength, viscosity, yield and Kappa number. Usually only Kappa number is measured on-line. The on-line measurement is located in the blow line of the digester. Thus, the main control actions are observed only after the delay time of the cooking and washing zones. The delay time can be several hours. By the prediction of the Kappa number prior to the cooking zone more information is provided to be applied in accurate control actions. The prediction indicates changes in the blow line Kappa number. The main control of the pulp quality should be applied in the pulp digester. The control of the pulp quality is more challenging in the following subprocesses of the plant, if the pulp quality (Kappa number) is out of the good quality area after the digester.

The fault diagnosis is needed to ensure accurate quality control of the chemical processes. Diagnosis of the chemical processes are studied in many papers, see e.g. ((Dash et al., 2003) and (Qian et al., 2003)). In the field of chemical pulping, there are not too many publications concerning fault diagnosis. Puranen (Puranen, 1999) has formed a disturbance index for process operators to be able to observe faulty process situations. In that study measurements, means and deviations are combined by fuzzy logic. Diagnosis of the digester has been also studied in papers (Ahvenlampi et al., 2005) and (Tervaskanto et al., 2005).

In large industrial plants, every sub process has its own task and the entire plant is working properly if all the sub processes are functioning effectively. A faulty operation in one sub process usually changes the performance of the entire plant. Therefore, faults

have to be found as quickly as possible and decisions that stop the propagation of their effects have to be made (Blanke et al., 2003). Active Fault Tolerant Control (AFTC) detects and isolates possible faults in the system and also reconfigures the control law (Mahmoud et al., 2003). In paper (Simani and Patton, 2002), a robust model-based technique for the diagnosis of faults in a chemical process has been developed. The system consists of a fuzzy combination of Takagi-Sugeno models. Fault Detection and Identification (FDI) is then applied by using residual analysis and geometrical tests.

In this study a continuous kraft cooking application is investigated. Most of the kraft pulp is produced in the continuous digesters (Gullichsen, 2000). In a typical chemical pulping process, the pre-treated and penetrated wood chips are fed into the impregnation vessel and pulp digester where lignin is removed from the chips with the aid of chemical reactions. The main active variables for the Kappa number are temperature, alkali concentration, cooking (residence) time and the wood species. The main lignin removal takes place in the cooking zone in the digester, where the temperature is significantly higher than in the impregnation vessel.

Applied Kappa number model (Gustafson's Kappa number model (Gustafson et al., 1983)) is also used in the real-time Kappa number modeling. The results for the conventional cooking process are presented in paper (Rantanen et al., 2003) and in the Downflow Lo-Solids^{TM} cooking process in paper (Rantanen et al., 2005).

In earlier study by the authors (Ahvenlampi et al., 2006), the proposed system was applied for the conventinal kraft cooking process. In this paper, the same approach is tested with Downflow Lo-Solids^{TM} kraft cooking process.

The aim of this approach has been to improve the Kappa number control by combining diagnosis (Ahvenlampi and Kortela, 2005) and new control strategy for Kappa number control (Rantanen, 2006). The used monitoring method is self-organizing map (SOM) (Kohonen, 1997). The evaluation of the problems in the process is performed by quantization error. If the quantization error is notable, the Kappa number prediction has been stopped.

The blow line Kappa number is predicted before cooking zone. Thus, the cooking temperature can be controlled. In the Kappa number control strategy, the cooking temperature's set point is determined by using only the Kappa number model.

The structure of the paper is an following. The methods used are presented in chapter 2. The proposed fault tolerant control system is presented in the chapter 3. Case study is considered in chapter 4 and discussion and conclusions are shown in chapters 5 and 6.

2 METHODS USED

In this chapter, methods used are presented. Empirical and experimental methods were applied. Gustafson's Kappa number model is an empirical model for delignification. The monitoring method, SOM, (Kohonen, 1997) is also presented.

2.1 Gustafson's Kappa Number Model

Gustafson *et al.* (Gustafson et al., 1983) have derived a mathematical model consisting of a series of differential equations describing the combined diffusion and kinetics within a wood chip during the kraft pulping process.

The lignin removal in the impregnation vessel can be calculated using Gustafson's Kappa number model for the initial phase. The rate equation for the initial phase delignification is:

$$\frac{dL}{dt} = k_{il} e^{(17.5-8760/T)} L \tag{1}$$

where L is the lignin content at time t,
k_{il} is a species specific constant and
T is temperature in Kelvin.

The rate equation for the bulk phase (cooking zone) delignification is:

$$\begin{aligned} \frac{\partial L}{\partial t} &= k_{0bl} e^{(A_l - B_l/T)} \left[OH^-\right] L + \\ & \quad k_{1bl} e^{(A_2 - B_2/T)} \left[OH^-\right]^{0.5} \left[HS^-\right]^{0.4} L, \end{aligned} \tag{2}$$

where $[OH^-]$ is the hydroxyl ion concentration,
$[HS^-]$ is the hydrosulphide ion concentration and
k_{0bl}, k_{1bl}, A_1, A_2, B_1 and B_2 are species specific constants.

The relative reaction rate is higher in the bulk phase than in the other phases.

Residual delignification happens in the washing zone and it is formulated as:

$$\frac{\partial L}{\partial t} = k_{rl} e^{(19.64-10804/T)} \left[OH^-\right]^{0.7} L, \tag{3}$$

where k_{rl} is a species specific constant for residual delignification.

The relative rate decreases, and the effect of hydroxyl ion concentration decreases in the residual phase. Parameters k_x, A_x and B_x are presented in (Rantanen, 2006).

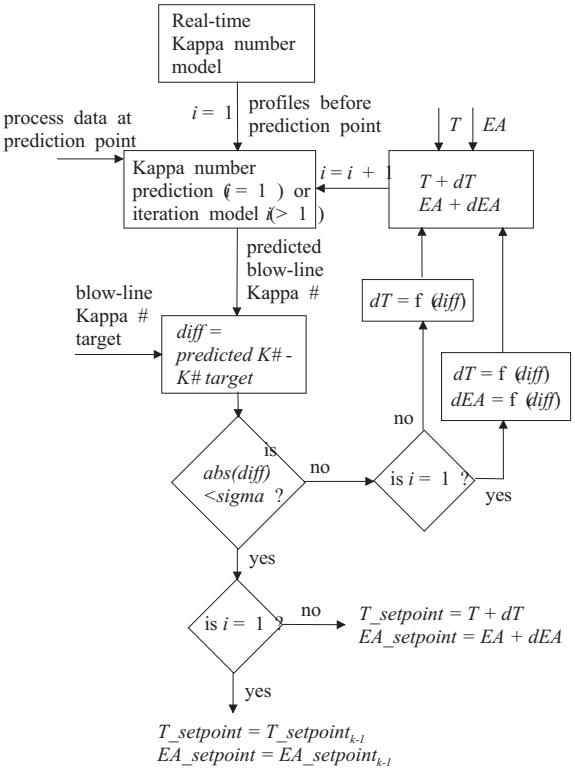


Figure 1: Structure of the new Kappa number control strategy.

2.2 Self-organizing Map

The SOM (Kohonen, 1997) is an unsupervised artificial neural network. The network is normally a two-dimensional mapping / projection of the data group. The visualization of the map is easier with a two-dimensional map. In the training of the SOM network, data points are sequentially introduced to the SOM. In each iteration, the SOM neuron which is closest to the input unit is selected by the equation (4). This unit is the Best Matching Unit (BMU) or winner.

$$\|z - c_c\| = \min_i \{\|z - c_i\|\} \qquad (4)$$

where z is input vector,

c_c is the selected center, Best Matching Unit (BMU), and

c_i is the current center in the evaluation.

The weight vectors are updated using the following formula. Only the weight vectors which are inside the neighborhood radius h_{ci}, are updated.

$$c_i(t+1) = c_i(t) + h_{ci}(t)[z(t) - c_i(t)] \qquad (5)$$

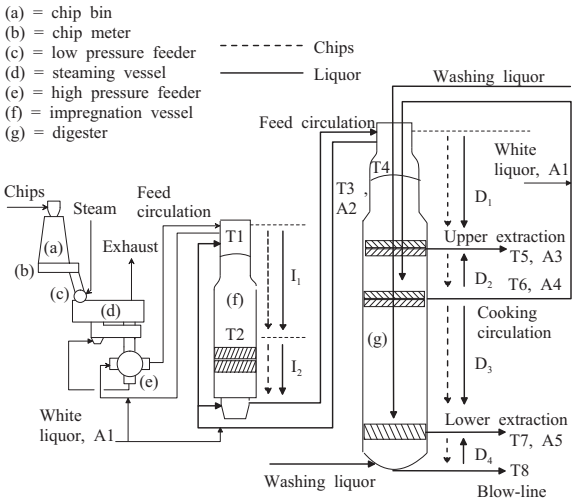


Figure 2: Impregnation vessel and continuous cooking digester.

3 KAPPA NUMBER CONTROL

In industrial plants, the Kappa number control is usually performed by the H-factor (Vroom, 1957). The H-factor expresses the cooking temperature and time as a single variable. Based on the difference between the predicted Kappa number and the Kappa number target the H-factor and temperature profile are corrected. One weakness of the H-factor is that depending on the variations in chip quality etc. different H-factors are needed, although the process conditions were otherwise the same. To overcame this problem new methods have been introduced. In this paper, Kappa number control strategy with and without a fault tolerant part is presented.

3.1 Kappa Number Control Strategy

New Kappa number control strategy is presented in study (Rantanen, 2006). In the approach, only Kappa number model is applied. That is the main difference compared to the use of Vroom's H-factor and Hatton's Kappa number model (Hatton, 1973). Thus, no separate models are needed to cover the effects of temperature, chemical concentrations and cooking time. The control of the alkali profile could also be improved by using the Kappa number model. In this paper, only temperature control is considered.

The procedure of the strategy is depicted in Figure 1. The blow line Kappa number is predicted before the cooking zone - in the middle of the Downflow Lo-Solids^{TM} digester (Figure 2). Predicted temperature and alkali profiles of the cooking zone, and online modelled Kappa number profile and process data

Table 1: Variables for the monitoring system before cooking zone (BCZ).

Variable	Unit
Alkali concentration BCZ	g/l (Na_2O, EA)
Temperature BCZ	K
Production rate BCZ	adt/d
Kappa number BCZ	

before cooking zone are used as inputs to the Kappa number prediction model. New temperature set point is solved iteratively based on the difference between the predicted and target blow line Kappa numbers. Also other process conditions, especially alkali profile, can be more precisely taken into account in the applied strategy.

3.2 Fault Tolerant Kappa Number Control Strategy

The fault tolerant control system (see structure in Figure 3) is formulated by the combination of diagnosis and control of the Kappa number in the continuous cooking plant. The self-organizing map (SOM) is applied for the monitoring purposes. The SOM is trained with normalized data. The inputs for the monitoring system are presented in Table 1. The quantization errors are used in the coloring of the trends of the measured inputs and the predicted Kappa number.

With the aid of the diagnosis part the problematic process conditions and measurement failures can be detected. If the process is not in the good operation area, the control of the Kappa number can be stopped and keep in the current state for the period of the poor operation. This ensures that the corrections are not done into wrong directions and there is not too strong corrections (overshoot).

Without monitoring problems can occur, due to the residence time between the cooking temperature control point (before cooking zone) and on-line Kappa number measurement. Inaccurate control can also bring other difficulties into the process. One of the problems is faulty packing in the digester. Faulty packing can occur, if the chips are cooked too long or too short a time. Due to the problems, also the shutdown of the process is possible.

4 CASE STUDY

Case study is Downflow Lo-Solids^{TM} Kamyr process consisting of an impregnation vessel and a steam / liquor phase digester (Figure 2). The chips are impregnated in the impregnation vessel (I1-I2) and in the

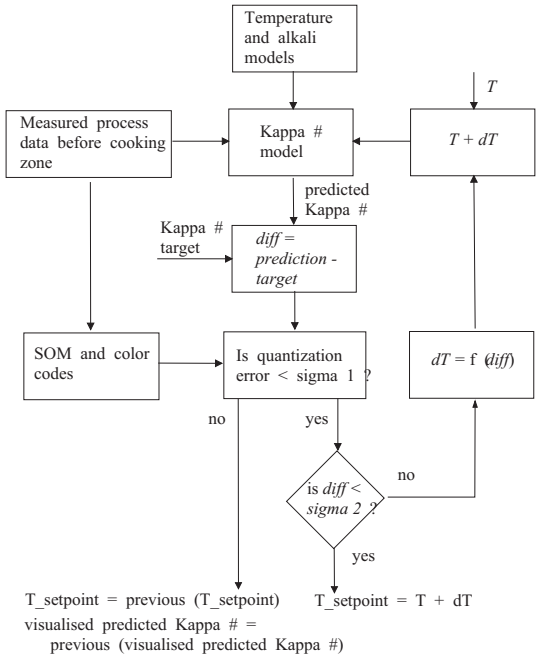


Figure 3: Structure of the fault tolerant Kappa number control system.

first zone (D1) of the digester. Between upper extraction and cooking circulation there is a counter-current washing zone (D2). In this zone, black liquor is displaced with cooking circulation liquor which temperature and alkali concentration are high. The lignin is mainly removed in the comparatively long co-current cooking zone (D3). At the bottom of the digester there is a short washing zone. Softwood chips mainly consist of pine chips with a small amount of spruce chips. Hardwood chips consist mainly of birch chips with a small addition of aspen chips.

In Downflow Lo-Solids^{TM} cooking, the Kappa number control is mainly performed by the cooking zone temperature and alkali in the middle of the digester (prior to D3).

In this study, fault tolerant control system is used for monitoring and control purposes in Downflow Lo-Solids^{TM} continuous cooking digester. The inputs to the system are monitored and the blow line Kappa number is controlled. The blow line Kappa number is predicted at the middle of the digester by using prediction model (Gustafson's Kappa number model) and the new temperature setpoint is calculated. The study is carried out in Matlab environment using measurements from the industrial continuous kraft cooking application.

The monitoring is carried out by using SOM. The modeling data (about one month data) was collected from the industrial continuous digester during its normal operation. The outliers and faulty measurements

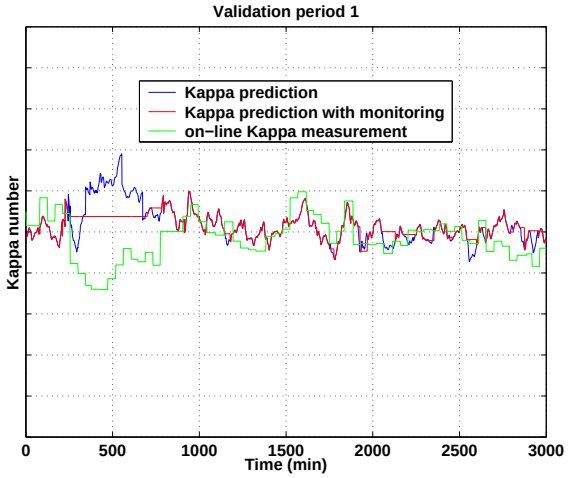


Figure 4: Validation period 1.

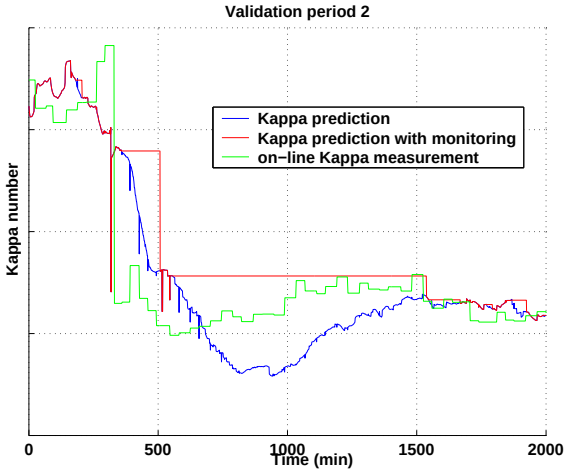


Figure 5: Validation period 2.

are filtered out from the data. The inputs are temperature, alkali, lignin content (Kappa number) and production rate before the cooking zone. The system is validated with the data from the same industrial digester, but from the different time periods.

In the Figures 4 - 6, are presented validation results of the proposed system. The monitoring system indicates, whether the prediction can be trusted or not. The stopping of the prediction is shown in Figure 4 in the time period 250-750.

Other example is shown in Figure 5, where the prediction after the grade change has not been good and it has shifted into wrong direction. The monitoring system has indicated problems in the process and the prediction has been stopped in time period 600-1500.

The same kind of example is shown in Figure 6. The prediction has been stopped for the period 600-1100.

5 DISCUSSION

The sampling interval of the on-line Kappa number measurements is about half an hour. Hence, it is useful to also get continuous information about quality properties. The accurate control can decrease significantly problems in the digester. The proposed system is a combination of diagnosis, prediction and control of Kappa number in the continuous kraft pulping digester. This kind of systems can be very helpfull for the operators.

The proposed fault tolerant control system gives new information for the control and the control actions are not taken into wrong directions as seen in Figure 4. In Figure 4 in time step 250-750, the prediction without monitoring is shifted dramatically too up and the control action would have been too strong into wrong direction. This has been avoided using monitoring and the stopping of the prediction.

In the Figures 5 and 6, are shown examples when there has been problems in the process after the grade change from softwood to hardwood. Diagnosis system has indicated problems and the Kappa number prediction has been stopped for these faulty periods. In the validaton period 2 (Figure 5), the prediction is stopped for the period 600-1500 and in the validation period 3 (Figure 6) for the period 600-1100.

The approach has been tested using the data from the industrial Downflow Lo-Solids^{TM} continuous cooking digester. Although the results are good, more research is needed to ensure the proper functioning of the proposed system in all operation points. Both prediction and diagnosis parts need some development in the future.

6 CONCLUSION

In this study, the combination of empirical and experimental methods for the diagnosis and monitoring of Kappa number control in Downflow Lo-Solids^{TM} cooking application was considered. Modeling and prediction of the Kappa number is applied using Gustafson's Kappa number model. The SOM is used to monitor the usability of the modelling and prediction results. The quality control is applied by controlling the temperature at the middle of the digester (before cooking zone) using diagnosis results and the prediction of the Gustafson's Kappa number model. Good results were achieved using the approach.

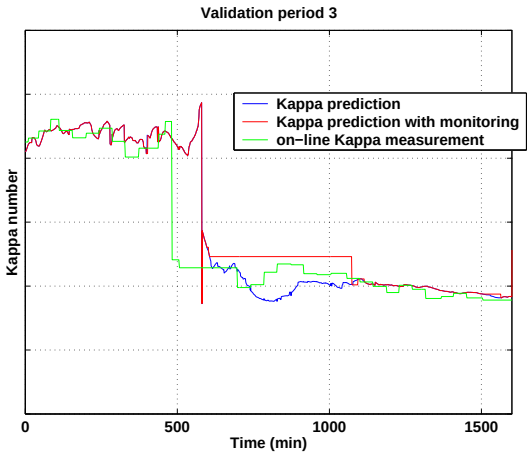


Figure 6: Validation period 3.

ACKNOWLEDGEMENTS

This research study was funded by The Acedemy of Finland, Metsä-Botnia Oy, Andritz Oy, Stora Enso Oyj, Metso Automation Oy and the Finnish Funding Agency for Technology and Innovation (Tekes). The authors would like to thank the partners for the special knowledge and process data provided.

REFERENCES

Ahvenlampi, T. and Kortela, U. (2005). Clustering algorithms in process monitoring and control application to continuous digesters. *Informatica*, 29(1):99–107.

Ahvenlampi, T., Rantanen, R., and Tervaskanto, M. (2006). Fault tolerant control application for continuous kraft pulping proces. In 6^{th} *IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes*, Beijing, PR China.

Ahvenlampi, T., Tervaskanto, M., and Kortela, U. (2005). Diagnosis system for continuous cooking process. In 16^{th} *IFAC World Congress*, Prague, Czech Republic.

Blanke, M., Kinneart, M., Lunge, J., and Staroswiecki, M. (2003). *Diagnosis and Fault-Tolerant Control*. Springer-Verlag, Berlin.

Dash, S., Rengaswamy, R., and Venkatasubramanian, V. (2003). Fuzzy-logic based trend classification for fault diagnosis of chemical processes. *Computers & Chemical Engineering*, 27:347–362.

Gullichsen, J. (2000). Chemical engineering principles of fiber line operations. In Gullichsen, J. and Fogelholm, C.-J., editors, *Chemical Pulping. Papermaking Science and Technology, Book 6A*. Fapet Oy, Jyväskylä, Finland.

Gustafson, R. G., Sleicher, C. A., McKean, W. T., and Finlayson, B. A. (1983). Theoretical model of the kraft pulping process. *Ind. and Eng. Chem., Process Design and Development*, 22(1):87–96.

Hatton, J. V. (1973). Development of yield prediction equations in kraft pulping. *Tappi J.*, 56(7):97–100.

Isermann, R. (1997). Supervision, fault-detection and fault-diagnosis methods - an introduction. *Control Engineering Practice*, 5(5):639–652.

Kohonen, T. (1997). *Self-Organizing Maps*. Springer-Verlag, Berlin, Germany, second edition.

Leiviskä, K. (2000). *Process Control*. Papermaking Science and Technology, Book 14. Fapet Oy, Helsinki, Finland.

Mahmoud, M., Jiang, J., and Zhang, Y. (2003). *Active Fault Tolerant Control Systems*. Lecture Notes in Control and Information Sciences. Springer-Verlag, Berlin, Germany.

Puranen, T. (1999). Jatkuvatoimisen sellukeittimen häiriöntunnistus sumealla logiikalla (in finnish). In *Automation'99 -Seminar*, pages 406–411, Helsinki, Finland.

Qian, Y., Li, X., Jiang, Y., and Wen, Y. (2003). An expert system for real-time fault diagnosis of complex chemical processes. *Expert Systems with Applications*, 24:425–432.

Rantanen, R. (2006). *Modelling and control of cooking degree in conventional and modified continuous pulping processes*. PhD thesis, University of Oulu, Oulu.

Rantanen, R., Ahvenlampi, T., and Kortela, U. (2003). Kappa number profile in continuous cooking - applying Gustafson's model to softwood and hardwood pulping process. In 4^{th} *Biennial Johan Gullichsen Colloquium*, pages 83–92, Espoo, Finland.

Rantanen, R., Similä, E., and Ahvenlampi, T. (2005). Modeling of kappa number in Downflow Lo-Solids^{TM} cooking using Gustafson's model. *Pulp & Paper Canada*, 106(5).

Simani, S. and Patton, R. J. (2002). Model-based data-driven approach to robust fault diagnosis in chemical processes. In *15th Triennial World Congress*, Barcelona, Spain. IFAC.

Tervaskanto, M., Ahvenlampi, T., Rantanen, R., and Kortela, U. (2005). Fault diagnosis of continuous cooking process using intelligent sensors. In *NeCST Workshop*, pages 187–192, Ajaccio, France.

Venkatasubramanian, V., Rengaswamy, R., and Kavuri, S. N. (2003a). A review of process fault detection and diagnosis part II: Qulitative models and search strategies. *Computers & Chemical Engineering*, 27:313–326.

Venkatasubramanian, V., Rengaswamy, R., Yin, K., and Kavuri, S. N. (2003b). A review of process fault detection and diagnosis part I: Quantitative model-based methods. *Computers & Chemical Engineering*, 27:293–311.

Vroom, K. E. (1957). The h-factor, a means of expressing cooking times and temperatures as a single variable. *Pulp. Pap. Mag. Can.*, 58(3):228–231.

NEW RESULTS ON DIAGNOSIS BY FUZZY PATTERN RECOGNITION

Mohamed Saïd Bouguelid, Moamar Sayed Mouchaweh and Patrice Billaudel
Centre de Recherche en Science et Technologie de l'Information (CReSTIC),Université de Reims-Champagne-Ardennes
Moulin de la Housse, BP 1039, 51687 Reims, France
ms.bouguelid@ univ-reims.fr, moamar.sayed-mouchaweh@ univ-reims.fr, patrice.billaudel@univ-reims.fr

Keywords: Pattern recognition, Fuzzy pattern matching, Nearest Neighbours techniques, Multi-criteria decision.

Abstract: We use the classification method Fuzzy Pattern Matching (FPM) to realize the industrial and medical diagnosis. FPM is marginal, i.e., its global decision is based on the selection of one of the intermediate decisions. Each intermediate decision is based on one attribute. Thus, FPM does not take into account the correlation between attributes. Additionally, FPM considers the shape of classes as convex one. Finally the classes are considered as equi-important by FPM. These drawbacks make FPM unusable for many real world applications. In this paper, we propose improving FPM to solve these drawbacks. Several synthetic and real data sets are used to show the performances of the Improved FPM (IFPM) with respect to classical one as well as to the well known classification method K Nearest Neighbours (KNN). KNN is known to be preferment in the case of data represented by correlated attributes or by classes with different a priori probabilities and non convex shape.

1 INTRODUCTION

In statistical Pattern Recognition (PR) (Dubuisson, 2001), historical patterns about system functioning modes are divided into groups of points, called classes, using unsupervised learning method (Duda, 2001) or human experience. These patterns, or points, with their class assignments, constitute the learning set. A supervised learning method uses the learning set to build a classifier that best separates the different classes in order to minimize the misclassification error. This separation, or classification, is realized by using a membership function, which determines the likelihood or the certainty that a point belongs to a class.

The membership function can be generated using Probability Density Function (PDF) estimation based methods or heuristics-based ones. In the first category, the membership function is equal to either the PDF or to the probability a posterior. The estimation of PDF can be parametric, as the baysien classifier (Dubuisson, 1990), or non parametric, as the Parzen window (Dubuisson, 2001), voting k nearest neighbour rules (Denoeux, 1998), (Denoeux, 2001) and (Dubuisson, 2001), or by histograms (Sayed Mouchaweh, 2004), (Medasani, 1998). In heuristic-based methods (Medasani, 1998), the shape of the membership function and its parameters are predefined either by experts to fit the given data set, or by learning to construct directly the decision boundaries as the potential functions (Dubuisson, 1990) and neural networks (Ripley, 1996), or the clustering methods as Fuzzy C-Means (Medasani, 1998).

One of the applications of PR is the diagnosis of industrial systems for which no mathematical or analytical information is available to construct a model about the system functioning. Each functioning mode, normal or faulty, is represented by a class. The problem of diagnosis by PR becomes a problem of classification, i.e., the actual functioning mode can be determined by knowing the class of the actual pattern, or observation, of the system functioning state.

There are many fuzzy classification methods in the literature. The choice of one of them depends on the given application and the available data. We use the method Fuzzy Pattern Matching (FPM) (Devillez, 2004b), (Grabish, 1992) and (Sayed Mouchaweh, 2002a) because it is simple, adapted to incomplete database cases and has a small and constant classification time. FPM is a marginal classification method, i.e., its global decision is

based on the selection of one of the intermediate decisions. Each intermediate decision is calculated using a membership function based on a probability histogram for each class according to each attribute. Thus, FPM is not adapted to work with data represented in a space of correlated attributes. Additionally, it does not respect the shape of classes if this shape is non convex. Finally, all the classes are considered as equi-important, i.e., with the same a priori probabilities.

In this paper we propose a solution to develop FPM to take into account the correlation between attributes as well as the class importance and its shape if this shape is non convex. The paper is structured as follows. Firstly, the functioning of FPM is explained briefly. Then the limits of FPM are discussed using some synthetic examples. Next, several real examples are used to evaluate the performances of the Improved FPM (IFPM) with respect to the classical one as well as to the well known classification method K Nearest Neighbours (KNN). This evaluation is based on the misclassification rate. Finally a conclusion ends this paper.

2 FUZZY PATTERN MATCHING

FPM, described in (Devillez, 2004b), (Grabish, 1992) and (Sayed Mouchaweh, 2002a), is a supervised classification method based on the use of probability histograms. Let C_1, C_2, …, C_c denote the classes described by a attributes. These attributes provide different points of view about the membership of an incoming point in the different classes. The functioning of FPM involves two phases: the learning and the classification ones.

2.1 Learning Phase

In the learning phase, the data histograms are constructed for each class according to each attribute. The number of bins h for a histogram is experimentally determined. This number has an important influence on the performances of FPM (Sayed Mouchaweh, 2002b). The histogram upper and lowest bounds can be determined either as the maximal and minimal learning data coordinates or by experts. In this paper, we adopted the first manner and we have added a tolerance *Tol* to adjust this histogram in order to maximize FPM performances. The height of each bin is the number of learning points located in this bin. The probability distribution p_i^j of the class C_i according to the attribute j is calculated by dividing the height of each bin b_k by the total number N_i of learning points belonging to the same class. Then these probabilities $\{p_i^j(y_{b_k}), k \in \{1,2,...,h\}\}$ are assigned to bins centres y_{b_k}. The PDF P_i^j is obtained by a linear linking between the bins heights centres. Indeed if we have a large number of data, the normalized histogram can be assumed to approximate the PDF.

In order to take into account the uncertainty and the imprecision contained in the data, the probability distribution p_i^j is converted into possibility one π_i^j. This conversion is realized using the transformation of Dubois and Prade (Dubois, 1993) defined as:

$$\pi_i^j(y_{b_k}) = \sum_{d=k}^{h} min(p_i^j(y_{b_k}), p_i^j(y_{b_d})) \qquad (1)$$

We have chosen this transformation for the good results which it gives in PR applications (Sayed Mouchaweh, 2002b), (Sayed Mouchaweh, 2006). A linear linking between bins heights centres converts the distribution of possibilities $\{\pi_i^j(y_{b_k}), k \in \{1,2,...,h\}\}$ into density one Π_i^j. This operation is repeated for all the attributes of each class.

2.2 Classification Phase

The membership function μ_i^j for each class C_i and according to each attribute j is considered to be numerically equivalent to the possibility distribution (Zadeh, 1978). Thus, the classification of a new point x, whose values of the different attributes are x^1, …, x^j, …, x^a, is made in two steps:

- Determination of the possibility membership value π_i^j of the point x to each class C_i according to the attribute j by a projection on the corresponding possibility density Π_i^j;
- Merging all the possibility values π_i^1, ..., π_i^a, concerning the class C_i, into a single one by an aggregation operator H:

$$\pi_i = H(\pi_i^1, ..., \pi_i^a) \qquad (2)$$

The result π_i of this fusion corresponds to the global possibility value that the new point x belongs to the class C_i. The operator H can be a

multiplication, a minimum, an average, or a fuzzy integral (Grabish, 1992). Finally, the point x is assigned to the class for which it has the maximum membership value.

3 LIMITS OF FPM

3.1 Classes of Non Convex Shape

FPM is similar to the naive Bayesian classifier who supposes the attributes statistically independent. This classifier defines the probability a posterior $\Phi(C_i|x)$ that x belongs to the class C_i by:

$$\Phi(C_i|x) = \prod_{j=1}^{a} p(x^j|C_i)P(C_i) \qquad (3)$$

Where $p(x^j|C_i)$ is the marginal conditional density of the attribute j given the class C_i and $P(C_i)$ denotes the a priori probability of the class C_i. If the a priori probabilities of classes are the same, then the equation (3) becomes similar to the equation (2). Thus FPM, as all the other marginal methods, does not take into account neither the correlation between attributes nor the class importance or its shape if it is non convex. Indeed, FPM produces always rectangular membership level curves for all the possible class shapes. Figure 1.a and Figure 1.b present respectively the membership level curves obtained by FPM for a class defined either by two linear or non linear correlated attributes. We can notice that these levels do not respect the class shape.

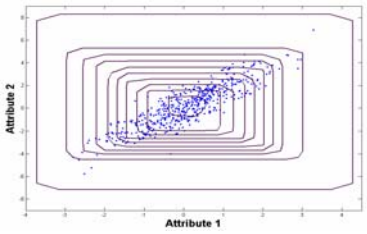


Figure 1.a: Membership level curves obtained by FPM for a class defined by linear correlated attributes.

In (Devillez, 2004a), two improvements were proposed to integrate the information about class shape in the learning phase. These two improvements are based on the division of each class into several sub ones. However these improvements have some drawbacks as the critical determination of the value of some parameters, like the number of subclasses, the expensive computation time and the rejection areas inside classes, i.e., areas in which points are not assigned to any class.

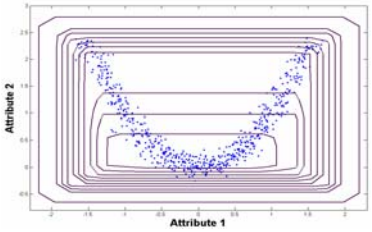


Figure 1.b: Membership level curves obtained by FPM for a class defined by non linear correlated attributes.

3.2 Classes with Correlated Attributes

The XOR data are a classical example used in the literature to show the correlation between attributes. Indeed any classifier needs to use the information issued from all the attributes to take a correct decision. Thus FPM is not adapted for this type of data since its decision is based on the selection of one attribute. The Figure 2 shows XOR data in a representation space of two attributes as well as the membership level curves, obtained by FPM for the class 1. We can see that the classes have a convex shape and that FPM does not distinguish between the points of the class 1 and the ones of the class 2. Thus the improvements proposed by Devillez (Devillez, 2004a) cannot solve this problem since they were developed to be adapted to the class shape and not to the case of correlated attributes. Same remark can be noticed for the membership level curves of the class 2.

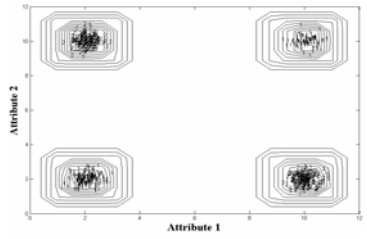


Figure 2: The membership level curves, obtained by FPM, for XOR data for the class 1.

In (Cadenas, 2004) a solution to make FPM operant in the case of data with correlated attributes is presented. This solution uses the Parzen Window method to construct the membership functions of each class according to one main feature and, if it is necessary, to one auxiliary feature. The

classification phase uses the fuzzy integral as an aggregation operator. However, this solution is consuming of time and has an exponential complexity according to the number of attributes. In addition, this solution does not work in the case of XOR described by more than two attributes.

4 IMPROVED FPM (IFPM)

We propose a solution to make FPM operant in the case of data with correlated attributes and classes of non convex shape. This solution looks for the relationship between the attributes of the representation space using the learning set. This relationship is represented by a correlation matrix between the bins of the histogram of the first attribute and all the other bins of the other histograms of the other attributes. This solution does not require any determination of any supplementary parameter. The functioning of IFPM is divided into two phases: learning and classification ones.

4.1 Learning Phase

The learning phase of IFPM is similar to the one of FPM but it integrates in addition the information about the zones of learning points inside the representation space. Each zone is resulting by the intersection of the bins of a learning point according to all attributes.

Let *X* denotes the learning set which contains *N* points *x* divided into *c* classes inside a representation space of *a* attributes. Each class C_i contains n_i points: $i \in \{1,2,...,c\}$. Each histogram for each attribute j, $j \in \{1,2,...,a\}$, contains h bins $b_{k_j}^j$, $k_j \in \{1,2,...,h\}$. The correlation matrix B for the learning set *X* is defined as follows:

$$B = [B^1,\ B^2,...,B^i,...,B^c] \qquad (4)$$

Where B^i is the correlation matrix for the class C_i. This matrix can be calculated as follows:

$$B^i = [\alpha_{b_x}^i \in \{1,0\}] \qquad (5)$$

Where *x* belongs to the zone resulting by the intersection of the bins $b_x = [b_{k_1}^1, b_{k_2}^2, ..., b_{k_a}^a]$. $\alpha_{b_x}^i$ is the correlation factor between the bin $b_{k_1}^1$ of the first attribute and all the other bins of the other attributes according to the class C_i. This correlation factor can be calculated using the following equation:

$$\forall x \in X, C(x) = C_i :$$
$$\begin{cases} \text{If } x \in b_{k_1}^1 \cap b_{k_2}^2 \wedge b_{k_1}^1 \cap b_{k_3}^3 \wedge ... \wedge b_{k_1}^1 \cap b_{k_a}^a \Rightarrow \alpha_{b_x}^i = 1 \\ \text{Else} \qquad \Rightarrow \alpha_{b_x}^i = 0 \end{cases} \qquad (6)$$

Where "$\wedge$" and "$\cap$" denote respectively the AND and intersection operators.

The Figure 3 shows a simple example of the calculation of the matrix *B* for the case of a representation space containing one class defined by two attributes. We can notice that the bins b_2^1 and b_2^2 are correlated because a learning point is located in the zone resulting by the intersection of these two bins. Thus the correlation factor of this zone $\alpha_{b_2^1 b_2^2}^1$ is equal to 1 in the matrix *B*. While there is no learning point in the zone of the intersection of the bins b_1^1 and b_5^2. The correlation factor of this zone $\alpha_{b_1^1 b_5^2}^1$ is equal to zero in *B*.

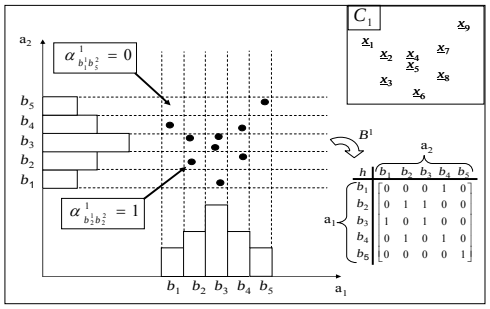


Figure 3: Correlation Matrix obtained by IFPM for the class C_1 inside $\Re^2$.

4.2 Classification Phase

The classification of a new point *x* starts by determining its bins according to each attribute $b_x = [b_{k_1}^1, b_{k_2}^2, ..., b_{k_a}^a]$. Then the possibility membership value π_i for each class C_i is calculated exactly as FPM if and only if $\alpha_{b_x}^i = 1$. In order to take onto account the importance of a class, the possibility of each class is multiplied by its a priori probability. If $\alpha_{b_x}^i = 0$, the point *x* will be rejected according to the class C_i, i.e., $\pi_i = 0$. Finally the point *x* will be assigned to the class for which it has the highest possibility membership value. The Figure 4

represents the steps of the classification phase of IFPM

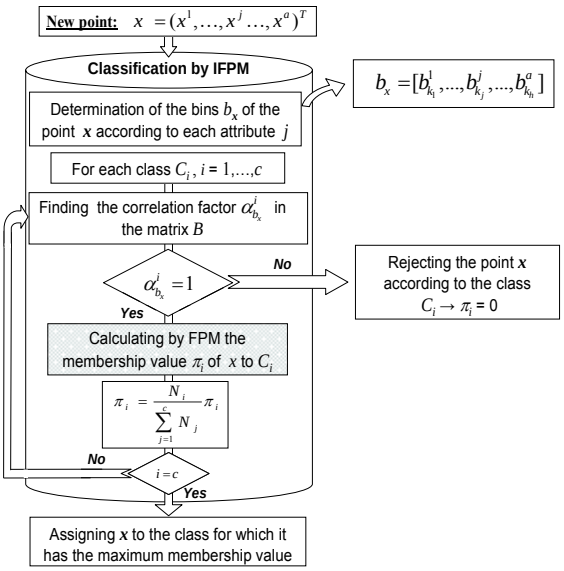


Figure 4: Algorithm of the classification phase of IFPM.

The Figure 5.a and Figure 5.b show the membership level curves issued from the application of IFPM on the two examples of the Figure 1.

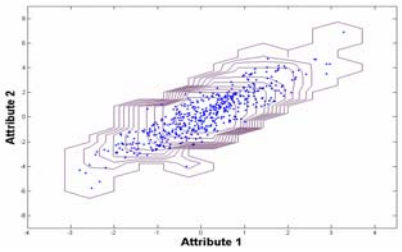


Figure 5.a: Membership level curves obtained by IFPM for a class defined by linear correlated attributes.

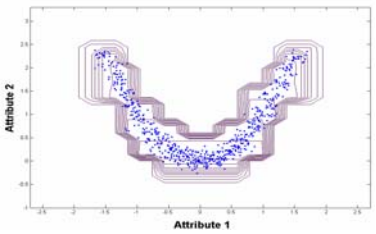


Figure 5.b: Membership level curves obtained by IFPM for a class defined by non linear correlated attributes.

We can find that these curves respect the non spherical and non convex shapes of classes.

Equally, the Figure 6 shows these curves for the class 1 obtained by IFPM, for XOR data. We can find that IFPM discriminates well the points of the class 1 and the class 2. Same result can be obtained for the class 2.

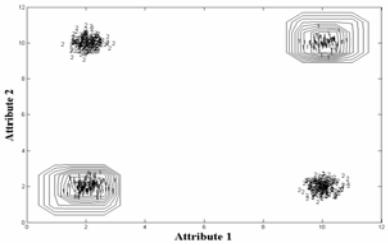


Figure 6: Membership level curves, obtained by IFPM, for XOR data for class 1.

5 EXPERIMENTAL RESULTS

We will test the performances of IFPM with FPM and classical as well as Fuzzy KNN (FKNN) with crisp initialisation and several values of k between 1 and 15, using the misclassification rate as evaluation criterion. For that we use four data sets: XOR problem, Spiral, Pima Indians diabetes (Newman, 1998) and Ljubljana Breast Cancer (LBC) data sets (Newman, 1998). XOR problem data set is composed of 2 classes in a representation space of 5 attributes. We have chosen 5 attributes to test IFPM performances in the case of a representation space characterized by more than 2 correlated attributes. Spiral data are represented in terms of evenly spaced samples from a non linear two-dimensional transformation of the Cartesian coordinates. We have chosen this data set because the classes are non linearly separable. The Pima and LBC data sets are known to be strongly non Gaussian. The number of points in each class, the number of classes and the number of attributes are depicted in Table 1.

We have used the leave-one-out method to calculate the Misclassification Rate (MR) because it gives a pessimistic unbiased estimation of MR. We integrate also the Rejection Rate (RR) to indicate the number of points which are not assigned to any class. Indeed, it is better to reject a point than to misclassify it. The Table 2 shows the obtained results for the three methods. These results are obtained using the optimal values of *h* and *Tol* for FPM and IFPM, and *K* for KNN. We can conclude that IFPM provides better results, according to the evaluation criterion and for the four data sets, than FPM and both, KNN and FKNN.

Table 1: Data Sets used for the test.

Data set	Classes	DIM	Per Class
XOR	2	5	{400,400}
Spiral	2	2	{970,970}
Pima	2	8	{268,500}
LBC	2	9	{218,68}

Table 2: Comparison between IFPM, FPM, KNN and FKNN, using leave-one-out technique, according to Rejection Rate (RR) and to Misclassification Rate (MR).

Method	**FPM**		**IFPM**	
Criterion	MR (%)	RR (%)	MR (%)	RR (%)
XOR	44.12	0	0	0
Spiral	18.24	0	0	1.5
Pima	30.73	0.78	19.53	20.7
LBC	25.87	1.05	10.14	30.42
Method	**KNN**		**FKNN**	
XOR	0	0	0	0
Spiral	0	0	0	0
Pima	25.26	0	26.43	0
LBC	23.78	0	23.43	0

6 CONCLUSIONS

In this paper, we have proposed a solution to adapt the classification method Fuzzy Pattern Matching (FPM) to be operant in the case of classes with correlated attributes as well as the class importance and its shape if this shape is not convex. The integration of this solution in FPM is called Improved FPM (IFPM). The performances of IFPM are compared with the ones of FPM, K Nearest Neighbours (KNN) and Fuzzy KNN (FKNN) using the misclassification rate as evaluation criterion. This comparison is realized according to four data sets. We have also used XOR problem and Spiral data which are widely used to study the correlation between attributes. In addition, we have used Pima Indians diabetes and Ljubljana Breast Cancer data sets which are known to be strongly non Gaussian with different a priori probabilities. The misclassification rate obtained by IFPM is better than the one of FPM, KNN and FKNN. However, IFPM rejects more points than the previous methods. Anyway, it is better to reject a point than to misclassify it.

REFERENCES

Cadenas, J.M., M.C. Garrido and J.J. Hernandez, 2004. Improving fuzzy pattern matching techniques to deal with non discrimination ability features. In: *IEEE International Conference on Systems, Man and Cybernetics*, pp. 5708-5713.

Denoeux, T., M. Masson and B. Dubuisson, 1998. Advanced pattern recognition techniques for system monitoring and diagnosis: a survey. In: *Journal Européen des Systèmes Automatisés* (RAIRO-APII-JESA), Vol. 31(9-10), pp. 1509-1539.

Denoeux, T. and L.M. Zouhal, 2001. Handling possibilistic labels in pattern classification using evidential reasoning. In: *Fuzzy Sets and Systems*, Vol. 1(22), pp. 409-424.

Devillez, A., 2004a. Four supervised classification methods for discriminating classes of non convex shape. In: *Fuzzy sets and systems*, Vol. 141, pp. 219-240.

Devillez, A., M.Sayed Mouchaweh and P. Billaudel, 2004b. A process monitoring module based on fuzzy logic and Pattern Recognition. In: *International Journal of Approximate Reasoning*, Vol. 37, Issue 1, pp.43-70.

Dubois, D. and H. Prade, 1993. On possibility/probability transformations, In: *Fuzzy Logic*, pp. 103-112.

Duda, R.O., P.E. Hart and D.E. Stork, 2001. *Pattern Classification second edition*. Wiley, New York

Dubuisson, B., 2001. *Automatique et statistiques pour le diagnostic*. Traité IC2 Information, commande, communication. Hermes Sciences, Paris.

Dubuisson, B., 1990. *Diagnostic et reconnaissance des formes*, Traité des Nouvelles Technologies, série Diagnostic et Maintenance, Hermes Sciences, Paris.

Grabish, M. and M. Sugeno, 1992. Multi-attribute classification using fuzzy integral. In: *Proc. of fuzzy IEEE*, pp. 47-54.

Medasani, S., K. Jaeseok and R. Krishnapuram, 1998, An overview of membership function generation techniques for pattern recognition. In: *International Journal of Approximate Reasoning*,Vol. 19,pp. 391-417.

Newman D.J., Hettich, S., Blake C.L. and Merz, C.J, 1998. UCI Repository of machine learning databases, http://www.ics.uci.edu/~mlearn/MLRepository.html, *Dept. of Information and Computer Science*, University of California, Irvine.

Ripley, B.D.,1996. *Pattern Recognition and Neural Networks*. Cambridge University Press, Cambridge.

Sayed Mouchaweh, M., A. Devillez, G.V. Lecolier and P. Billaudel, 2002a. Incremental learning in real time using fuzzy pattern matching. In: *Fuzzy Sets and Systems*, Vol. 132/1, pp. 49-62.

Sayed Mouchaweh, M. and P. Billaudel ,2002b. Influence of the choice of histogram parameters at Fuzzy Pattern Matching performance. In: WSEAS Transactions on Systems, Vol. 1, Issue 2, pp. 260-266

Sayed Mouchaweh, M., 2004. Diagnosis in real time for evolutionary processes in using Pattern Recognition and Possibility theory. In: *International Journal of Computational Cognition 2*, Vol. 1, pp. 79-112.

Zadeh, L. A., 1978. Fuzzy sets as a basis for a theory of possibility. In: *Fuzzy sets and systems*, Vol. 1, pp. 3-2.

IMITATING THE KNOWLEDGE MANAGEMENT OF COMMUNITIES OF PRACTICE

Juan Pablo Soto[1], Aurora Vizcaíno[1], Javier Portillo[1], Oscar M. Rodríguez-Elias[2] and Mario Piattini[1]
[1]*Alarcos Research Group, Information Systems and Technologies Department, UCLM-Soluziona Research and Development Institute, University of Castilla – La Mancha, Spain*
jpsoto@proyectos.inf-cr.uclm.es, {aurora.vizcaino, mario.piattini}@uclm.es, javier.portillo@alu.uclm.es
[2]*UABC, Facultad de Ciencias, Ensenada México*
orodrigu@cicese.mx

Keywords: Reputation, multi-agent architecture, communities of practice, knowledge management.

Abstract: Advances in technology have led to the development of knowledge management systems with the intention of improving organizational performance. Nevertheless, implementation of this kind of mechanisms is not an easy task due to the necessity of taking into account social aspects (such as reputation) that improve the exchange of information between groups of people. Considering, the advantages of working with groups with similar interests we have modelled communities of agents which represent communities of people interested in similar topics. In order to implement this model we propose a multi-agent architecture in charge of evaluating the relevance of the knowledge in a knowledge base and the degree of reputation that a person has as the contributor of information. We pay particular attention to showing how the use of the agents works by using a prototype system to search for knowledge related to a particular domain of a community of practice. Several communities of agents integrated into an organization have the capacity to follow the interaction process of employees when carrying out their daily activities.

1 INTRODUCTION

For several decades human behaviour has been studied with the objective of imitating certain aspects of it in computational systems (Schaeffer et al, 1996). Based on this idea we have studied how the people obtain and increase their knowledge in their daily work. From this study we realise that frequently, employees exchange knowledge with people who work on similar topics as them and consequently, either formally or informally, communities are created which can be called "communities of practice", by which we mean groups of people with a common interest where each member contributes knowledge about a common domain (Wenger, 1998).

Communities of practice enable their members to benefit from each other's knowledge. This knowledge resides not only in people's minds but also in the interaction between people and documents. An interesting fact is that individuals are frequently more likely to use knowledge built by their community team members than those created by members outside their group (Desouza et al, 2006). This factor occurs because people trust more in the information offered by a member of their community than in that supplied by a person who does not belongs to that community. Thus, a new concept takes place in the process of obtaining information. This concept is "trust" and can be defined as "confidence in the ability and intention of an information source to deliver correct information" (Barber & Kim, 2004). Therefore, people, in real life in general and in companies in particular, prefer to exchange knowledge with "trustworthy people" by which we mean people they trust. Of course, the fact of belonging to the same community of practice already implies that they have similar interests and perhaps the same level of knowledge about a topic. Consequently, the level of trust within a community is often higher that which exists out of the community. Because of this, as is claimed in (Desouza et al, 2006), knowledge reuse tends to be restricted within groups.

Bearing in mind that people exchange information with "trustworthy knowledge sources" we have designed a multi-agent architecture in which agents try to emulate humans evaluating

knowledge sources with the goal of fostering the use of knowledge bases in companies where agents provide "trustworthy knowledge" to the employees. Thus, in section 2 the multi-agent architecture is described. Then, in section 3 we illustrate how the architecture has been used to implement a prototype which detects and suggests trustworthy documents for members in a community of practice. In section 4 related works are outlined. Finally in section 5 conclusions are described.

2 A MULTI-AGENT ARCHITECTURE TO DEVELOP TRUSTWORTHY KNOWLEDGE BASES

Many organizations worried about their competitive advantage use knowledge bases to store their knowledge. However, sometimes the knowledge which is put into a system is not very valuable. This decreases the trust that employees have in their organizational knowledge and reduces the probability that people will use it. In order to avoid this situation we have developed a multi-agent architecture in charge of monitoring and evaluating the knowledge that is stored in a knowledge base.

To design this architecture we have taken into account how people obtain information in their daily lives. Bearing in mind the advantages of working with groups of similar interests we have organized the agents into communities of people who are interested in similar topics. Thus, Figure 1 shows different communities where there are two types of agents: the *User Agent* and the *Manager Agent*. The former is used to represent each person that may consult or introduce knowledge in a knowledge base. The *User Agent* can assume three types of behaviour or roles similar to the tasks that a person can carry out in a knowledge base. Therefore, the User Agent plays one role or another depending upon whether the person that it represents carries out one of the following actions:

- The person contributes new knowledge to the communities in which s/he is registered. In this case the User Agent plays the role of **Provider (Pr)**.
- The person uses knowledge previously stored in the community. Then, the User Agent will be considered as a **Consumer (Co)**.
- The person helps other users to achieve their goals, for instance by giving an evaluation of certain knowledge. In this case the role is of a **Partner (Pa)**. So, Figure 1 shows that in community 1 there are two User Agents playing the role of Partner, one User Agent playing the role of Consumer and another being a Provider.

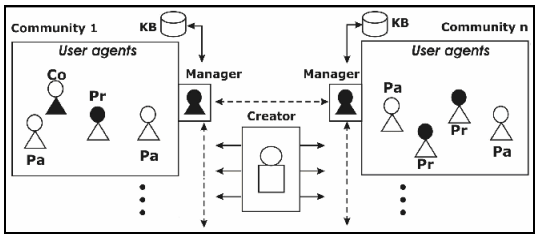


Figure 1: Multi-agent architecture.

The fact that this agent can act both as consumers and also as providers of knowledge may lead to better results because they aim to motivate the active participation of the individual in the learning process, which often results in the development of creativity and critical thinking (Kan, 1999).

The conceptual model of this agent, whose goals are to detect trustworthy agents and sources, is based on two closely related concepts: trust and reputation. The former can be defined as confidence in the ability and intention of an information source to deliver correct information (Barber & Kim, 2004) and the latter as the amount of trust an agent has in an information source, created through interactions with information sources. This definition is the most appropriate for our research since the level of confidence in a source is based on previous experience of this. It is for this reason that the remainder of the paper deals solely with reputation. However, if we attempt to imitate the behaviour of the employees in a company when they are exchanging and obtaining information we observe that apart from the concept of reputation other factors also influence. These are:

- Position: employees often consider information that comes from a boss as being more reliable than that which comes from another employee in the same (or a lower) position as him/her (Wasserman & Glaskiewics, 1994). However, this is not a universal truth and depends on the situation. For instance in a collaborative learning setting collaboration is more likely to occur between people of a similar status than between a boss and his/her employee or between a teacher and pupils (Dillenbourg, 1999). Because of this, as will be explained

later, in our research this factor will be calculated by taking into account a weight that can strengthen this factor to a greater or to a lesser degree.

- Expertise: this term can be briefly defined as the skill or knowledge of a person who knows a great deal about a specific thing. This is an important factor since people often trust in experts more than in novice employees. Moreover, tools such as expertise location (Crowder et al, 2002) are being developed with the goal of promoting the sharing of expertise knowledge (Rodríguez-Elias et al, 2004).
- Previous experience: People have greater trust in those sources from which they have previously obtained more "valuable information". Therefore, a factor that influences the increasing or decreasing reputation of a source is "previous experience" and this factor can help us to detect invaluable sources or knowledge. One problem occurring in organizations is that some employees introduce information which is not particularly useful in a knowledge base with the only objective of trying to simulate that they are contributing information in order to generate points or benefits such as incentives or rewards (Huysman & Wit, 2000). When this happens, the information stored is generally not very valuable and it will probably never be used.

Taking all these factors into account we have defined an own "concept of reputation" (see Figure 2).

Figure 2: Reputation module.

That is, the reputation of $agent_s$ about $agent_i$ is a collective measure defined by the previously describe factors and computed as follows:

$$R_{si} = w_e * E_i + w_p * P_i + \left(\sum_{i=1}^{n} QC_i\right)/n$$

where R_{si} denotes the reputation value that $agent_s$ has in $agent_i$ (each agent in the community has an opinion about each one of the agent members of the community) .

w_e and w_p are weights with which the Reputation value can be adjusted to the needs of the organizations.

E_i is the value of expertise which is calculated according to the degree of experience that a person has in a domain.

P_i is the value assigned to the position of a person. This position is defined by the organizational diagram of the enterprise. Therefore, a value that determines the hierarchic level within the organization can be assigned to each level of the diagram.

In addition, previous experience should also be calculated. To accomplish this it is supposed that when an agent A consults information from another agent B, the agent A should evaluate how useful this information was. This value is called QC_i (Quality of *i*'s Contribution). To attain the average value of an agent's contribution, we calculate the sum of all the values assigned to their contributions and we divide it between their total. In the formula n represents the total number of evaluated contributions.

In this way, an agent can obtain a value related to the reputation of another agent and decide to what degree it is going to consider the information obtained from this agent.

The second type of agent within a community is called the *Manager Agent* (represented in black in Figure 1) which is in charge of managing and controlling its community. In order to approach this type of agent the following tasks are carried out:

- Registering an agent in its community. It thus controls how many agents there are and how long the stay of each agent in that community is.
- Registering the frequency of contribution of each agent. This value is updated every time an agent makes a contribution to the community.
- Registering the number of times that an agent gives feedback about other agents' knowledge. For instance, when an agent "A" uses information from another agent "B", the agent A should evaluate this information. Monitoring how often an agent gives feedback about other agents' information helps to detect whether agents contribute to the creation of knowledge flows in the community since it is as important that an agent contributes with new information as it is that another agent contributes by evaluating the relevance or importance of this information.
- Registering the interactions between agents. Every time an agent evaluates the contributions of another agent the Manager agent will register this interaction. But this

interaction is only in one direction, which means, if the agent A consults information from agent B and evaluates it, the Manager records that A knows B but that does not means that B knows A because B does not obtain any information about A.

Besides these agents there is another in charge of initiating new agents and creating new communities. This agent has two main roles: the "creator" role is assumed when there is a petition (made by a User Agent) to create a new Community and the "initiator" role is assumed when the system is initially launched. This agent, which is not included in any of the communities, is located in the centre of Figure 1, and is called the *Creator Agent*.

3 USING THE ARCHITECTURE

In order to evaluate the architecture and to gradually improve it we have developed a prototype system into which people can introduce documents and where these documents can also be consulted by other people. The goal of this prototype is that agents software help employees to discover the information that may be useful to them thus decreasing the overload of information that employees often have and strengthening the use of knowledge bases in companies. In addition, we try to avoid the situation of employees storing valueless information in the knowledge base.

The main feature of this system is that when a person searches for knowledge in a community, and after having used the knowledge obtained, that person then has to evaluate the knowledge in order to indicate whether:

- The knowledge was useful
- How it was related to the topic of the search (for instance a lot, not too much, not at all).

User Agents will use this information to construct a "trust net". Thus, these agents can know how reliable the contributions of each person are and also what each contribution was. This information is very important to companies since by consulting it, it is possible to know which employees are the best contributors. From this information other information can be obtained. For instance, who should be consulted when there is a problem in a concrete domain, since we agree with (Ackoff, 1989) who claim that knowledge management systems should encourage dialogue between individuals rather than simply directing them to repositories.

In the next sub-sections, we describe different situations or scenarios to show how the agents work in this prototype. These situations will represent some general community rules and will show the main interactions between agents in a community.

3.1 A New User Arrives in a Community

This situation happens when, for instance, a user wants to join to a new community. To do this, the person will choose a community from all the available communities. In this case the Manager Agent will ask whether there is any agent that knows the new user in order to set a trust value on this person (this process is similar to the socialization stage of the SECI model (Nonaka & Takeuchi, 1995), where each one indicates its experience about a topic, in this case about another person).

When a user wants to join to a community in which no member knows anything about him/her, the reputation value assigned to the user in the new community is calculated on the basis of the reputation assigned from others communities where the user is or was a member. In order to do this, the User Agent called, for instance, j, will ask each community manager where he/she was previously a member to consult each agent which knows him/her with the goal of calculating the average value of his/her reputation (R_j). This is calculated as:

$$R_j = \left(\sum_{i=1}^{n} R_{ij}\right)/n$$

where n is the number agents who know j and R_{ij} is the value of j's reputation in the eyes of i. In the case of being known in several communities the average of the values R_j will be calculated. Then, the User Agent presents this reputation value (similar to when a person presents his/her curriculum vitae when s/he wishes to join in a company) to the community manager to which it is "applying". In the case of the user being new in the system (s/he has never been in a community) then this user is assigned a "new" label in order for the situation to be identified.

Once the Community Manager has obtained a Reputation value for j it is added to the community member list.

3.2 Using Community Documents and Updating Reputation Values

People can search for documents in every community in which they are registered. When a

person searches for a document relating to a topic his/her User Agent consults the Manager Agent about which documents are related to their search. Then, the Manager agent answers with a list of documents. The User Agent sorts this list according to the reputation value of the authors, which is to say that the contributions with the best reputations for this Agent are listed first. On the other hand, when the user doesn't know the contributor then the User Agent consults the Manager Agent about which members of the community know the contributors. Thus, the User Agent can consult the opinions that other agents have about these contributors, thus taking advantage of other agents' experience. To do this the Manager consults its interaction table and responds with a list of the members who know the User Agent Then, this User Agent contacts each of them. If nobody knows the contributors then the information is listed, taking their expertise and positions into account. In this way the User Agent can detect how worthy a document is, thus saving employees' time, since they do not need to review all documents related to a topic but only those considered most relevant by the members of the community or by the person him/herself according to previous experience with the document or its authors.

Once the person has chosen a document, his/her User Agent adds this document to its own document list (list of consulted documents), and if the author of the document is not known by the person because it is the first time that s/he has worked with him/her, then the Community Manager adds this relation to the interaction table explained in section 2. This step is very important since when the person evaluates the document consulted, his/her User Agent will be able to assign a QC for this document.

4 RELATED WORK

This research can be compared with other proposals that use agents and trust communities in knowledge exchange. In literature we found several trust and reputation mechanisms that have been proposed for large open environments, for instance: e-commerce (Zachaira et al, 1999), peer-to-peer computing (Wang & Vassileva, 2003), etc.

There are others works on trust and reputation (Griffiths, 2005; Yu & Singh, 2000). We shall only mention those works that are most related to our approach.

In (Schulz et al, 2003), the authors propose a framework for exchanging knowledge in a mobile environment. They use delegate agents to be spread out into the network of a mobile community and use trust information to serve as a virtual presence of a mobile user. Another interesting work is (Wang & Vassileva, 2003). In this work the authors describe a trust and reputation mechanism that allows peers to discover partners who meet their individual requirements through individual experience and sharing experiences with other peers with similar preferences. This work is focused on peer-to-peer environments.

Barber and Kim present a multi-agent belief revision algorithm based on belief networks (Barber & Kim, 2004). In their model the agent is able to evaluate incoming information, to generate a consistent knowledge base, and to avoid fraudulent information from unreliable or deceptive information source or agents. This work has a goal similar to ours. However, the means of attaining it are different. In Barber and Kim's case they define reputation as a probability measure, since the information source is assigned a reputation value of between 0 and 1. Moreover, every time a source sends knowledge the source should indicate the certainty factor that the source has of that knowledge. In our case, the focus is very different since it is the receiver who evaluates the relevance of a piece of knowledge rather than the provider as in Barber and Kim's proposal.

Therefore, the main difference between our work and previous works is that we take into account factors that might influence the level of trust that a person has in a piece of knowledge and in a knowledge source. Moreover, we present a general formula to define the reputation concept. This formula can be adapted, by modifying the value of the weights, to different settings. This is an important difference from other works which are focused on particular domains.

5 CONCLUSIONS

Communities of practice have the potential to improve organizational performance and facilitate community work. Because of this we consider it important to model people's behaviour within communities with the purpose of imitating the exchange of information in companies that are produced in those communities. Therefore, we attempt to encourage the sharing or information in organizations by using knowledge bases. To do this we have designed a multi-agent architecture where the artificial agents use similar parameters to those

of humans in order to evaluate knowledge and knowledge sources. These factors are: Reputation, expertise and of course, previous experience.

This approach implies several advantages for organizations as it permits them to identify the expertise of their employees and to measure the quality of their contributions. Therefore, it is expected that a greater flow of communication will exist between them which will consequently produce an increase in their knowledge.

In addition, this work has illustrated how the architecture can be used to implement a prototype. The main functionalities of the prototype are:

- Controlling whether employees try to introduce valueless knowledge with the goal of obtaining some profit such as points, incentives, rewards,, etc
- Providing the most suitable knowledge for the employee's queries, by using the reputation and relevance values that the agents have obtained from previous experiences.
- Detecting the expertise of the employees within an organization.

All these advantages provide organizations with a better control of their knowledge base which will have more trustworthy knowledge and it is consequently expected that employees will feel more willing to use it.

ACKNOWLEDGEMENTS

This work is partially supported by the ENIGMAS (PIB-05-058), and MECENAS (PBI06-0024) project, Junta de Comunidades de Castilla-La Mancha, Consejería de Educación y Ciencia, both in Spain. It is also supported by the ESFINGE project (TIN2006-15175-C05-05) Ministerio de Educación y Ciencia (Dirección General de Investigación)/ Fondos Europeos de Desarrollo Regional (FEDER) in Spain.

REFERENCES

Ackoff, R., 1989, *From Data to Wisdom.* Journal of Applied Systems Analysis, Vol. 16, pp: 3-9.

Barber, K., Kim, J., 2004, Belief Revision Process Based on Trust: Simulation Experiments. In *4th Workshop on Deception, Fraud and Trust in Agent Societies*, Montreal Canada.

Crowder, R., Hughes, G., Hall, W., 2002, Approaches to Locating Expertise Using Corporate Knowledge. *International Journal of Intelligent Systems in Accounting Finance & Management*, Vol. 11, pp: 185-200.

Desouza, K., Awazu, Y., Baloh, P., 2006, *Managing Knowledge in Global Software Development Efforts: Issues and Practices.* IEEE Software, pp: 30-37.

Dillenbourg, P., 1999, *Introduction: What Do You Mean By "Collaborative Learning"?.* Collaborative Learning Cognitive and Computational Approaches. Dillenbourg (Ed.). Elsevier Science.

Griffiths, N., 2005, *Task Delegation Using Experience-Based Multi-Dimensional Trust*, AAMAS'05, pp: 489-496.

Huysman, M., Wit, D., 2000, *Knowledge Sharing in Practice*. Kluwer Academic Publishers, Dordrecht.

Kan, G., 1999, *Gnutella.* Peer-to-Peer: Harnessing the Power of Disruptive Technologies. O'Reilly, pp: 94-122.

Nonaka, I., Takeuchi, H., 1995, *The Knowledge Creation Company: How Japanese Companies Create the Dynamics of Innovation*. Oxford University Press.

Rodríguez-Elias, O., Martínez-García, A., Favela, J., Vizcaíno, A., Piattini, M., 2004, *Understanding and Supporting Knowledge Flows in a Community of Software Developers.* LNCS 3198, Springer, pp: 52-66.

Schaeffer, J., Lake, R., Lu, P., Bryant, M., 1996, *CHINOOK: The World Man-Machine Checkers Champion.* AI Magazine, Vol. 17, pp: 21-29.

Schulz, S., Herrmann, K., Kalcklosch, R., Schowotzer, T., 2003, *Trust-Based Agent -Mediated Knowledge Exchange for Ubiquitous Peer Networks.* AMKM, LNAI 2926, pp: 89-106.

Wang, Y., Vassileva, J., 2003, *Trust and Reputation Model in Peer-to-Peer Networks.* Proceedings of IEEE Conference on P2P Computing.

Wasserman, S., Glaskiewics, J., 1994, *Advances in Social Networks Analysis.* Sage Publications.

Wenger, E., 1998, *Communities of Practice: Learning Meaning, and Identity*, Cambridge U.K.: Cambridge University Press.

Yu, B., Singh, M., 2000, *A Social Mechanism of Reputation Management in Electronic Communities.* Cooperative Information Agents, CIA-2000, pp: 154-165.

Zachaira, G., Moukas, A., Maes, P., 1999, *Collaborative Reputation Mechanisms in Electronic Marketplaces.* 32nd Annual Hawaii International Conference on System Science (HICSS-32).

THE VERIFICATION OF TEMPORAL KNOWLEDGE BASED SYSTEMS
A Case-study on Power-systems

Jorge Santos, Zita Vale, Carlos Ramos
Instituto Superior de Engenharia do Porto, Rua Dr. Antonio Bernardino de Almeida, 431, 4200-072 Porto – Portugal
{ajs,zav,csr}@isep.ipp.pt

Carlos Serôdio
Departamento de Engenharias, Universidade de Trás–os–Montes e Alto Douro, 5001-801 Vila Real – Portugal
cserodio@utad.pt

Keywords: Knowledge–based Systems Engineering, Verification, Temporal Reasoning, Power Systems.

Abstract: Designing KBS for dynamic environments requires the consideration of temporal knowledge reasoning and representation (TRR) issues. Although humans present a natural ability to deal with knowledge about time and events, the codification and use of such knowledge in information systems still pose many problems. Hence, the development of applications strongly based on temporal reasoning remains an hard and complex task. Furthermore, albeit the last significant developments in TRR area, there is still a considerable gap for its successful use in practical applications.

This paper presents a tool, named VERITAS, developed for temporal KBS verification. It relies in the combination of formal methods and heuristics, in order to detect a large number of knowledge anomalies. The underlying verification process addresses many relevant aspects considered in real applications, like the usage of rule triggering selection mechanisms and temporal reasoning.

1 INTRODUCTION

The methodologies proposed in software engineering showed to be inadequate for knowledge based systems (KBS) validation and verification, since KBS present some particular characteristics (Gonzalez and Dankel, 1993).

In last decades knowledge based systems became a common tool in a large number of power systems control centres (CC) (Kirschen and Wollenberg, 1992). In fact, the number, diversity and complexity of KBS increased significantly leading to important changes in KBS structure. Designing KBS for dynamic environments requires the consideration of TRR (Temporal Reasoning and Representation) issues. Although humans present a natural ability to deal with knowledge about time and events, the codification and use of such knowledge in information systems still pose many problems. Hence, the development of applications strongly based on temporal reasoning remains an hard and complex task. Furthermore, in despite of the last significant developments in TRR area, there is still a considerable gap for its successful use in practical applications.

This paper addresses the verification of knowledge based systems through the combination of formal methods and heuristics. The rest of the paper is organized as follows: the section 2 provides a short overview of the state-of-art of V&V and its most important concepts and techniques. Section 3 , describes the the study case, SPARSE, a KBS used to assist the Portuguese Transmission Control Centres operators in incident analysis and power restoration. Section 4 introduces the problem of verifying real world applications. Finally, section 5 presents VERITAS, describing the methods used to detect knowledge anomalies.

2 RELATED WORK

In the last decades several techniques were proposed for validation and verification of knowledge based systems, including for instance, inspection, formal proof, cross–reference verification or empirical tests (Preece, 1998), regarding that the efficiency of these techniques strongly depends on the existence of test cases or in the degree of formalization used on the specifications. One of the most used techniques is static verification, that consists of sets of logical

tests executed in order to detect possible knowledge anomalies. An anomaly is a symptom of one, or more, possible error(s). Notice that an anomaly does not necessarily denotes an error (Preece and Shinghal, 1994).

Some well known V&V tools used different techniques to detect anomalies. The KB–Reducer (Ginsberg, 1987) system represents rules in logical form, then it computes for each hypothesis the corresponding labels, detecting the anomalies during the labeling process. Meaning, that each literal in the rule LHS (Left Hand Side) is replaced by the set of conditions that allows to infer it. This process finishes when all formulas became grounded. The COVER (Preece et al., 1992) works in a similar fashion using the ATMS (Assumption Truth Maintaining System) approach (Kleer, 1986) and graph theory, allowing to detect a large number of anomalies. The COVADIS (Rousset, 1988) successfully explored the relation between input and output sets.

The systems ESC (Cragun and Steudel, 1987), RCP (Suwa et al., 1982) and Check (Nguyen et al., 1987) and later, Prologa (Vanthienen et al., 1997) used decision table methods for verification purposes. This approach proved to be quite interesting, specially when the systems to be verified also used decision tables as representation support. These systems major advantage is that tracing reasoning path becomes quite clear. But in the other hand, there is a lack of solutions for verifying long reasoning inference chains.

Some authors studied the applicability of Petri nets (Pipard, 1989; Nazareth, 1993) to represent the rule base and to detect the knowledge inconsistencies. Colored Petri nets were later used (Wu and Lee, 1997). Although specific knowledge representations provide higher efficiency while used to perform some verification tests, arguably all of them could be successful converted to production rules.

Albeit there is no general agreement on the V&V terminology, the following definitions will be used for the rest of this paper.

Validation means building the right system (Boehm, 1984). The purpose of validation is to assure that a KBS will provide solutions with similar (or higher if possible) confidence level as the one provided by domain experts. Validation is then based on tests, desirably in the real environment and under real circumstances. During these tests, the KBS is considered as a *black box*, meaning that, only the input and the output are really considered important.

Verification means building the system right (Boehm, 1984). The purpose of verification is to assure that a KBS has been correctly designed and implemented and does not contain technical errors. During the verification process the interior of the KBS is examined in order to find any possible errors, this approach is also called *crystal box*.

3 SPARSE – A CASE STUDY

Control Centers (CC) are very important in the operation of electrical networks receiving real–time information about network status. CC operators should take, usually in a short time, the most appropriate actions in order to reach the maximum network performance.

Concerning SPARSE , in the beginning it started to be an expert system (ES) and it was developed for the Control Centers of the Portuguese Transmission Network (REN). The main goals of this ES were to assist Control Center operators in incident analysis allowing a faster power restoration. Later, the system evolved to a more complex architecture (Vale et al., 2002), which is normally referred as a knowledge based system (see Fig.1).

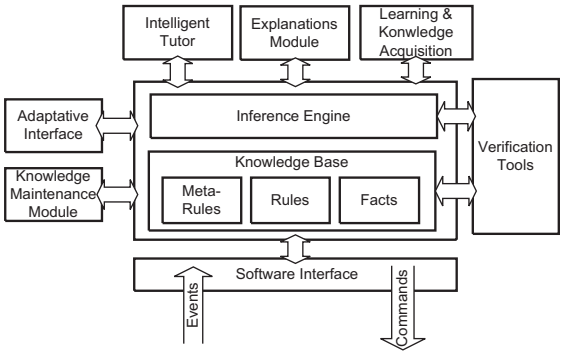


Figure 1: SPARSE Architecture.

One of the most important components of SPARSE is the knowledge base (KB) (see the formula 1):

$$KB = RB \cup FB \cup MRB \quad (1)$$

where:

- RB stands for rule base;
- FB stands for facts base;
- MRB stands for meta–rules base;

The rule base is a set of Horn clauses with the following structure:

```
RULE ID: 'Description':
[
 [C1 AND C2 AND C3]
 OR
```

```
 [C4 AND C5]
]
==>
[A1,A2,A3].
```

The rule's LHS (Left Hand Side) is a set of conditions (C1 to C5 in this example) of the following types:

- A fact, representing domain events or status messages. Typically these facts are time–tagged;
- A temporal condition over facts;
- Previously asserted conclusions

The actions/conclusions to be taken in RHS (Right Hand Side) (A1 to A3 in this example) may be of one of the following types:

- Assertion of facts (conclusions to be inserted in the knowledge base);
- Retraction of facts (conclusions to be deleted from the knowledge base);
- Interaction with the user interface.

The meta–rule base is a set of triggers, used by rule selection mechanism, with the following structure:

$$trigger(Fact, [(R_1, Tb_1, Te_1), \ldots, (R_n, Tb_n, Te_n)])$$

standing for:

- *Fact* – the arriving fact (external alarm or a previously inferred conclusion);
- $(Rule_x, Tb_1, Te_1)$ – the tuple is composed by: $Rule_x$, the rule that should be triggered in when fact arrives; Tb_1 the delay time before rule triggering, used to wait for remaining facts needed to define an event; Te_1 the maximum time for trying to trigger the each rule.

The inference process relies on the cycle. In the first step, SPARSE collects one *message* from SCADA[1], then the respective *trigger* is selected and some rules are scheduled. The temporal window were the rule X could by triggered is defined in the interval $[Tb, Te]$. The *scheduler* selects the next rule to be tested, (the inference engines tries to prove its veracity). Notice that, when a rule succeeds, the conclusions (on the RHS) will be asserted and later processed in the same way as the SCADA messages.

[1]Supervisory Control And Data Acquisition, this systems collects messages from the mechanical/electrical devices installed in the network

4 THE VERIFICATION PROBLEM

Traditionally the verification problem through anomaly detection relies on the computation of all possible inference chains (expansions) that could be entailed during the reasoning process. Later, some logical tests are performed in order to detect if any constraints violation takes place (see Fig.2 core).

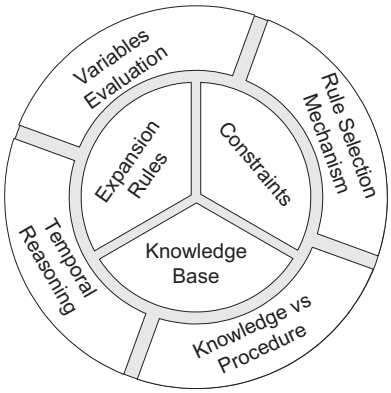


Figure 2: The verification problem.

SPARSE presents some features, namely, temporal reasoning and rule triggering mechanism (see Fig.2), that makes the verification work harder. These features demand the use of more complex techniques during anomaly detection and introduce significant changes in the number and type of anomalies to detect.

4.1 Rule Triggering Selection Mechanism

In what concerns SPARSE, this mechanism was implemented using both meta–rules and the inference engine. When a *message* arrives, some rules are selected and scheduled in order to be later triggered and tested.

In what concerns verification work, this mechanism not only avoids some run–time errors (for instance circular chains) but also introduces another complexity axis to the verification. Thus, this mechanism constrains the existence of inference chains and also the order that they would be generated. For instance, during system execution, the inference engine could be able to assure that shortcuts (specialists rules) would be preferred over generic rules.

4.2 Temporal Reasoning

This issue received large attention from scientific community in last two decades (surveys covering this issue can be found in (Gerevini, 1997; Fisher et al., 2005)). Despite the fact that *time* is ubiquitous in the society and the natural ability that human beings show dealing with it, a widespread representation and usage in the artificial intelligence domain remains scarce due to many philosophical and technical obstacles. SPARSE is an *alarm processing application* and its major challenge is to reasoning about events. Thus, it is necessary to deal with time intervals (*e.g.*, temporal windows of validity), points (*e.g.*, instantaneous events occurrence), alarms order, duration and the presence or/and absence of data (*e.g.*, messages lost in the collection/transmission system).

4.3 Variables Evaluation

In order to obtain comprehensive and correct results during the verification process, the evaluation of the variables present in the rules is crucial, especially in what concerns temporal variables, *i.e.*, the ones that represent temporal concepts. Notice that during anomaly detection (this type of verification is also called static verification) it is not possible to predict the exact value that a variable will have. Some techniques, concerning the variables domain and range, were considered in order to avoid the exponencial growth of the number of expansions during its computation.

4.4 Knowledge Versus Procedure

Languages like Prolog provide powerful features for knowledge representation (in the declarative way) but they are also suited to describe procedures, so, sometimes knowledge engineers encode rule using ”procedural” predicates. For instances, the following sentence in Prolog: `min(X,Y,Min)` calls a procedure that compares `X` and `Y` and instantiates `Min` with smaller value. So, is not an (pure) knowledge item, in terms of verification it should be evaluated in order to obtain the `Min` value. It means the verification method needs to consider not only the programming language syntax but also the meaning (semantic) in order to evaluate the functions. This step is particularly important for any variables that are updated during the inference process.

5 VERITAS

VERITAS's main goal is to detect and report anomalies, allowing the users decide whether reported anomalies reflect knowledge problems or not. Basically, anomaly detection consists in the computation of all possible inference chains that could be produced during KBS performance. Later, the inference chains are tracked in order to find out if some constraint were violated.

After a filtering process, which includes *temporal reasoning analysis* and *variable evaluation* the system decides when to report an anomaly and eventually suggest some repair procedure. VERITAS is knowledge domain and rule grammar independent, due to these properties (at least theoretically), VERITAS could be used to verify any rule–based systems. Let's consider the following set of rules:

$$r1 : st1 \wedge ev1 \rightarrow st2 \wedge st3$$
$$r2 : st3 \wedge ev3 \rightarrow st6 \wedge ev5$$
$$r3 : ev3 \rightarrow ev4$$
$$r4 : ev1 \wedge ev2 \rightarrow ev4 \wedge st4$$
$$r5 : ev5 \wedge st5 \wedge ev4 \rightarrow st7 \wedge st8$$

VERITAS can show the rule dependencies through a directed hypergraph type representation (see Fig.3). This technique allows rule representation in a manner that clearly identifies complex dependencies across compound clauses in the rule base and there is a unique directed hypergraph representation for each set of rules.

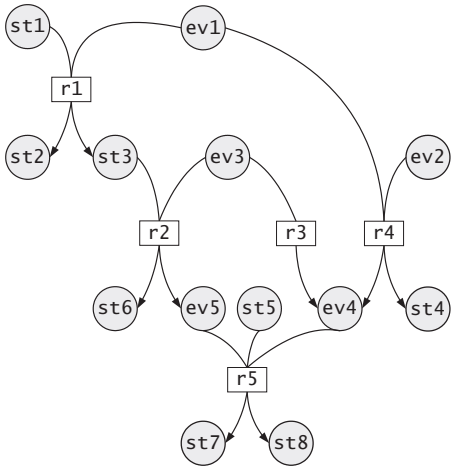


Figure 3: Hypergraph.

After the expansions calculation three different tuples are created, regarding ground facts, conclusions

and circular chains, with the following structure, respectively, $f(Fact)$, $e(SL,SR)$ and $c(SL,SR)$, where, Fact stands for a ground fact, SL stands for a set of conclusions and SR stands for a set of supporting rules.

So considering the example previously provided, the following two data sets would be obtained, since there are two possible expansions that allow to infer the labels st7 and st8:

```
f(ev3) f(st5) f(ev3) f(ev1) f(st1)
e([ev4],[[r3]])
e([st2,st3],[[r1]])
e([st6,ev5],[[r1,r2]])
e([st7,st8],[[r3,r5],[r1,r2,r5]])
```

```
f(ev2) f(ev1) f(st5) f(ev3) f(ev1) f(st1)
e([ev4,st4],[[r4]])
e([st2,st3],[[r1]])
e([st6,ev5],[[r1,r2]])
e([st7,st8],[[r4,r5],[r1,r2,r5]])
```

After the expansion computation, VERITAS detects a large set of anomalies. These anomalies can be grouped in three groups, circularity, ambivalence and redundancy, the anomalies classification used is based on Preece classification (Preece and Shinghal, 1994) with some modifications. Next sections describe some of the special cases.

5.1 Circularity

A knowledge base contains circularity if and only if it contains a set of rules, which allows an infinite loop during rule triggering. During the circularity detection, VERITAS considers the matching values in rule analysis, meaning that a new set of anomalies will arise. Let us consider the rules *rc1* and *rc2*:

$$rc1 : t(a) \wedge r(X) \rightarrow s(a)$$
$$rc2 : s(a) \rightarrow r(a)$$

Since a is a valid value for the argument X, some inference engines could start an infinite loop, so such circular inference chain should be reported. Another situation concerns temporal analysis through the use of heuristics. The rules *rc3* and *rc4* describes the turning on and off of a device D.

$$rc3 : st1(D,\mathsf{on},t1) \wedge ev1(D,\mathsf{turnoff}) \rightarrow st2(D,\mathsf{off},t2)$$
$$rc4 : st2(D,\mathsf{off},t1) \wedge ev2(D,\mathsf{turnon}) \rightarrow st1(D,\mathsf{on},t2)$$

If nothing else is stated, these two rules could "configure" a circular chain. So, before reporting an anomaly, VERITAS checks if none of the following situations happens, first, if the instant t2 is later than t1. Second, if in each rule LHS's appears an event (here represented by the ev label).

5.2 Redundancy

A knowledge base is redundant if and only if the set of final hypotheses is the same in the rule/literal presence or absence. A specific situation not considered in Preece anomaly classification concerns to redundancy between groups of rules. Consider the following example:

$$rr1 : a \wedge b \wedge c \rightarrow z$$
$$rr2 : \neg a \wedge c \rightarrow z$$
$$rr3 : \neg b \wedge c \rightarrow z$$

since the rules $rr1$, $rr2$ and $rr3$ are equivalent to the following logical expression:

$$rrx : (a \wedge b \wedge c) \vee (\neg a \wedge c) \vee (\neg b \wedge c) \rightarrow z$$

Applying logical simplifications to the previous rule, it is possible to obtain the following one:

$$rrx : c \rightarrow z$$

These situations are detected with an algorithm for logical expressions simplification. Basically, this algorithm works as follows:

1. for each existing conclusion, compute the set of LHSs of all rules that allows to infer it;
2. try simplify disjunction of the set obtained in the previous step;
3. compare the original expression with the simplified one, if the former is simpler (less variables or less terms) then report a redundancy anomaly;

This algorithm allows multi–valued logic, for instances, consider the following example, regarding that there are just three valid values for the *b* parameter's, respectively: open, closed and changing.

$$rr5 : a \wedge b(\mathsf{open}) \rightarrow z$$
$$rr6 : a \wedge b(\mathsf{closed}) \rightarrow z$$
$$rr7 : a \wedge b(\mathsf{changing}) \rightarrow z$$

the rules $rr5$, $rr6$ and $rr7$ are equivalent to the following rule:

$$rry : a \wedge (b(\mathsf{changing}) \vee b(\mathsf{closed}) \vee b(\mathsf{open})) \rightarrow z$$

Applying logical simplifications to the previous rule, it is possible to obtain the following one:

$$rry : a \rightarrow z$$

Notice that, in fact, redundancy between groups of rules is a generalization of the unused literal situation already studied by Preece.

5.3 Ambivalence

A knowledge base is ambivalent if and only if, for a permissible set of conditions, it is possible to infer an impermissible set of hypotheses. For ambivalence detection, VERITAS considers some types of constraints (also referred as *impermissible sets*), representing sets of contradictory conclusions. The constraints can be one following types:

Semantic Constraints – formed by literals that cannot be present at the same time in the KB. For instance, the installation i cant be controlled remotely and locally at same time.

$$\forall i,t1,t2 : \bot \leftarrow Remote(\mathsf{d},\mathsf{On},\mathsf{t1}) \wedge Local(\mathsf{d},\mathsf{Off},\mathsf{t2}) \wedge intersepts(t1,t2)$$

Single Value Constraints – formed by only one literal (Device) but considering different values of its parameters. For instance, the device d cant be on and off at same time.

$$\forall d,t1,t2 : \bot \leftarrow Device(\mathsf{d},\mathsf{On},\mathsf{t1}) \wedge Device(\mathsf{d},\mathsf{Off},\mathsf{t2}) \wedge intersepts(t1,t2)$$

The relation *intersepts* checks wether two intervals have some instant in common. Later, VERITAS computes the various expansions for each item present in a constraint and later determines if there exist a minimal set of facts that allow to infer contradictory conclusions/hypothesis and in such case an anomaly is reported.

6 CONCLUSIONS

This paper described VERITAS, a verification tool that successful combines formal methods, as structural analysis, and heuristics, in order to detect knowledge anomalies and provide useful reports. During its evaluation, the SPARSE was used as study case.

ACKNOWLEDGEMENTS

This work is partially supported by the Portuguese MCT-FCT project EDGAR (POSI/EIA/61307/2004). We would like to thanks the anonymous reviewers for their useful and detailed feedback.

REFERENCES

Boehm, B. (1984). Verifying and validating software requirements and design specifications. *IEEE Software*, 1(1):75–88.

Cragun, B. and Steudel, H. (1987). A decision table based processor for checking completeness and consistency in rule based expert systems. *International Journal of Man Machine Studies (UK)*, 26(5):633–648.

Fisher, M., Gabbay, D., and Vila, L., editors (2005). *Handbook of Temporal Reasoning in Artificial Intelligence*, volume 1 of *Foundations of Artificial Intelligence Series*. Elsevier Science & Technology Books.

Gerevini, A. (1997). Reasoning about time and actions in artificial intelligence: Major issues. In O.Stock, editor, *Spatial and Temporal Reasoning*, pages 43–70. Kluwer Academic Pblishers.

Ginsberg, A. (1987). A new aproach to checking knowledge bases for inconsistency and redundancy. *In proceedings 3rd Annual Expert Systems in Government Conference*, pages 10–111.

Gonzalez, A. and Dankel, D. (1993). *The Engineering of Knowledge Based Systems - Theory and Practice*. Prentice Hall International Editions.

Kirschen, D. S. and Wollenberg, B. F. (1992). Intelligent alarm processing in power systems. *Proceedings of the IEEE*, 80(5):663–672.

Kleer, J. (1986). An assumption-based TMS. *Artificial Intelligence Holland*, 2(28):127–162.

Nazareth, D. (1993). Investigating the applicability of petri nets for rule based systems verification. *IEEE Transactions on Knowledge and Data Engineering*, 4(3):402–415.

Nguyen, T., Perkins, W., Laffey, T., and Pecora, D. (1987). Knowledge Based Verification. *AI Magazine*, 2(8):69–75.

Pipard, E. (1989). Detecting inconsistencies and incompleteness in rule bases: the INDE system. In *Proceedigns of the 8th International Workshop on Expert Systems and their Applications, 1989, Avignon,France;*, volume 3, pages 15–33, Nanterre, France. EC2.

Preece, A. (1998). Building the right system right. In *Proc.KAW'98 Eleventh Workshop on Knowledge Acquisition, Modeling and Management*.

Preece, A., Bell, R., and Suen, C. (1992). Verifying knowledge-based systems using the cover tool. *Proccedings of 12th IFIP Congress*, pages 231–237.

Preece, A. and Shinghal, R. (1994). Foundation and application of knowledge base verification. *International Journal of Intelligent Systems*, 9(8):683–702.

Rousset, M. (1988). On the consistency of knowledge bases:the COVADIS system. In *Procedings of the European Conference on Artificial Intelligence (ECAI'88)*, pages 79–84, Munchen.

Suwa, M., Scott, A., and Shortliffe, E. (1982). An aproach to verifying completeness and consistency in a rule based expert system. *AI Magazine (EUA)*, 3(4):16–21.

Vale, Z., Ramos, C., Faria, L., Malheiro, N., Marques, A., and Rosado, C. (2002). Real-time inference for knowledge-based applications in power system control centers. *Journal on Systems Analysis Modelling Simulation (SAMS), Taylor&Francis*, 42:961–973.

Vanthienen, J., Mues, C., and Wets, G. (1997). Inter Tabular Verification in an Interactive Environment. *Proceedings of the European Symposium on the Verification and Validation of Knowledge Based Systems*, pages 155–165.

Wu, C.-H. and Lee, S.-J. (1997). Enhanced High-Level Petri Nets with Multiple Colors for Knowledge Verification/Validation of Rule-Based Expert Systems. *IEEE TRansactions on Systems. Man, and Cybernetics*, 27(5):760–773.

LINEAR PROGRAMMING FOR DATABASE ENVIRONMENT

Akira Kawaguchi and Jose Alfredo Perez
Department of Computer Sciences, The City College of New York, New York, New York 10031, U.S.A.
akira@cs.ccny.cuny.edu, alfredo.portes@gmail.com

Keywords: Linear programming, simplex method, revised simplex method, database, stored procedure.

Abstract: Solving large-scale optimization problems requires an integration of data-analysis and data-manipulation capabilities. Nevertheless, little attempt has been made to facilitate general linear programming solvers for database environments. Dozens of sophisticated tools and software libraries that implement linear programming model can be found. But, there is no database-embedded linear programming tool seamlessly and transparently utilized for database processing. The focus of this study is to fill out this kind of technical gap of data analysis and data manipulation, in the event of solving large-scale linear programming problems for the applications built on the database environment. Specifically, this paper studies the representation of the linear programming model in relational structures and the computational method to solve the linear programming problems. Foundations for and preliminary experimental results of this study are presented.

1 INTRODUCTION

Linear programming is a powerful technique for dealing with the problem of allocating limited resources among competing activities, as well as other problems having a similar mathematical formation (Winston, 1994; Richard, 1991; Walsh, 1985). It has become an important field of optimization in the areas of science and engineering. It has become a standard tool of great importance for numerous business and industrial organizations. In particular, large-scale linear programming problems arise in practical applications such as logistics for large spare-parts inventory, revenue management and dynamic pricing, finance, transportation and routing, network design, and chip design (Hillier and Lieberman, 2001).

While these problems inevitably involve the analysis of a large amount of data, there has been relatively little work in the context of database processing. Little attempt has been made to facilitate data-driven analysis with data-oriented techniques. In today's marketplace, dozens of sophisticated tools and software libraries that implement linear programming model can be found. Nevertheless, these products do not work with database systems seamlessly. They rather require additional software layers built on top of databases to extract and transfer data in the databases. The focus of our study gathered here is to fill out this kind of technical gap of data analysis and data manipulation, in the event of solving large-scale linear programming problems for the applications built on the database environment.

In mathematics, linear programming problems are optimization problems in which the objective function to characterize an optimality of a problem and the constraints to express specific conditions for that problem are all *linear* (Hillier and Lieberman, 2001; Thomas H. Cormen and Stein, 2001). Two families of solution methods, so-called *simplex methods* (Dantzig, 1963) and *interior-point methods* (Karmarkar, 1984), are in wide use and available as computer programs today. Both methods progressively improve series of trial solutions by visiting edges of the feasible boundary or the points within the interior of feasible region, until a solution is reached to satisfy the conditions for an optimum. In fact, it is known that large problem instances render even the best of codes nearly unusable (Winston, 1994). Furthermore, the program libraries available today are found outside the standard database environment, thus mandat-

ing the use of a special interface to interact with these tools for linear programming computations.

This paper presents a detailed account on the methodology and technical issues to realize a general linear programming method for the relational database environment. In general, relational database is not for matrix operations for solving linear programming problems. Indeed, realizing matrix operations on top of the standard relational structure is non-trivial. In this paper, implementation techniques and key issues for this development are studied extensively, and a model suitable to capture the dynamics of linear programming computations is incorporated into the aimed development, by way of realizing a set of procedural interfaces that enables a standard database language to define problems within a database and to derive optimal solutions for those problems without requiring users to write detailed program statements.

Specifically, we develop a set of ready to use *stored procedures* to solve general linear programming problems. A stored procedure is a group of SQL statements, precompiled and physically stored within a database (Gulutzan and Pelzer, 1999; Gulutzan, 2007). It forms a logical unit to encapsulate a set of database operations, defined with application program interface to perform a particular task, thereby having complex logic run inside the database via a stored procedure. The exact implementation of a stored procedure varies from one database to another, but is supported by most major database vendors. To this end, we will show an implementation using MySQL open-source database system.

As a summary, contributions of this paper are threefold: First, we present a detailed account on the methodology and technical issues to integrate a general linear programming method into relational databases. Second, we present the development as forms of stored procedures for today's representative database systems. Third, we present an experimental performance study based on a comprehensive system that implements all these concepts. This is to investigate applications of business decision problems to a database environment for practical use.

2 FUNDAMENTALS

Consider the matrix notation expressed in the set of equations (1) below. The standard form of the linear programming problem is to maximize an objective function $Z = \mathbf{c}^T\mathbf{x}$, subject to the functional constraints of $\mathbf{Ax} \leq \mathbf{b}$ and non-negativity constraints of $\mathbf{x} \geq \mathbf{0}$, with $\mathbf{0}$ in this case being the n-dimensional zero column vector. A coefficient matrix $\mathbf{A}$ and column vectors $\mathbf{c}$, $\mathbf{b}$, and $\mathbf{x}$ are defined in the obvious manner such that each component of the column vector $\mathbf{Ax}$ is less than or equal to the corresponding component of the column vector $\mathbf{b}$.

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \mathbf{c} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}, \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix},$$

$$\mathbf{0} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \quad (1)$$

The goal is to find an optimal solution, that is, the most favorable values of the objective function among feasible ones for which all the constraints are satisfied. The *simplex method* (Dantzig, 1963) is an algebraic iterative procedure where each round of computation involves solving a system of equations to obtain a new trial solution for the optimality test. The simplex method relies on the mathematical property that the objective function's maximum must occur on a corner of the space bounded by the constraints of the feasible region.

To apply the simplex method, linear programming problems must be converted into augmented form, by introducing non-negative *slack variables* to replace non-equalities with equalities in the constraints. The problem can then be rewritten on the following form:

$$\mathbf{x_s} = \begin{bmatrix} x_{n+1} \\ x_{n+2} \\ \vdots \\ x_{n+m} \end{bmatrix}, [\mathbf{A}, \mathbf{I}] \begin{bmatrix} \mathbf{x} \\ \mathbf{x_s} \end{bmatrix} = \mathbf{b}, \begin{bmatrix} \mathbf{x} \\ \mathbf{x_s} \end{bmatrix} \geq \mathbf{0},$$

$$\begin{bmatrix} 1 & -\mathbf{c}^T & \mathbf{0} \\ \mathbf{0} & \mathbf{A} & \mathbf{I} \end{bmatrix} \begin{bmatrix} Z \\ \mathbf{x} \\ \mathbf{x_s} \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{b} \end{bmatrix} \quad (2)$$

In equations (2) above, $\mathbf{x} \geq \mathbf{0}$, a column vector of slack variables $\mathbf{x_s} \geq \mathbf{0}$, and $\mathbf{I}$ is the $m \times m$ identity matrix. Following the convention, the variables set to zero by the simplex method are called *nonbasic variables* and the others are called *basic variables*. If all of the basic variables are non-negative, the solution is called *a basic feasible solution*. Two basic feasible solutions are adjacent if all but one of their nonbasic variables are the same. The spirit of the simplex method utilizes a rule for generating from any given basic feasible solution a new one differing from the old in respect of just one variable.

Thus, moving from the current basic feasible solution to an adjacent one involves switching one vari-

able from nonbasic to basic and vice versa for one other variable. This movement involves replacing one nonbasic variable (called *entering* basic variable) by a new one (called *leaving* basic variable) and identifying the new basic feasible solution. The simplex algorithm is summarized as follows:

Simplex Method:

1. Initialization: transform the given problem into an augmented form, and select original variables to be the nonbasic variables (*i.e.*, $\mathbf{x} = \mathbf{0}$), and slack variable to be the basic variables (*i.e.*, $\mathbf{x_s} = \mathbf{b}$).
2. Optimality test: rewrite the objective function by shifting all the nonbasic variables to the right-hand side, and see if the sign of the coefficient of every nonbasic variable is positive, in which case the solution is optimal.
3. Iterative Step:
   - 3.1 Selecting an entering variable: as the nonbasic variable whose coefficient is largest in the rewritten objective function used in the optimality test.
   - 3.2 Selecting a leaving variable: as the basic variable that reaches zero first when the entering basic variable is increased, that is, the basic variable with the smallest upper bound.
   - 3.3 Compute a new basic feasible solution: by applying the Gauss-Jordan method of elimination, and apply the above optimality test.

2.1 Revised Simplex Method

The computation of the simplex method can be improved by reducing the number of arithmetic operations as well as the amount of round-off errors generated from these operations (Hillier and Lieberman, 2001; Richard, 1991; Walsh, 1985). Notice that n nonbasic variables from among the $n+m$ elements of $\left[\mathbf{x}^T, \mathbf{x_s}^T\right]^T$ are always set to zero. Thus, eliminating these n variables by equating them to zero leaves a set of m equations in m unknowns of the basic variables. The spirit of the *revised simplex method* (Hillier and Lieberman, 2001; Winston, 1994) is to preserve the only pieces of information relevant at each iteration, which consist of the coefficients of the nonbasic variables in the objective function, the coefficients of the entering basic variable in the other equations, and the right-hand side of the equations.

Specifically, consider the equations (3) below. The revised method attempts to derive a basic (square) matrix $\mathbf{B}$ of size $m \times m$ by eliminating the columns corresponding to coefficients of nonbasic variables from $[\mathbf{A}, \mathbf{I}]$ in equations (2). Furthermore, let $\mathbf{c_B}^T$ be the vector obtained by eliminating the coefficients of nonbasic variables from $\left[\mathbf{c}^T, \mathbf{0}^T\right]^T$ and reordering the elements to match the order of the basic variables. Then, the values of the basic variables become $\mathbf{B}^{-1}\mathbf{b}$ and $Z = \mathbf{c_B}^T\mathbf{B}^{-1}\mathbf{b}$. The equations (2) become equivalent with equations (3) after any iteration of the simplex method.

$$\mathbf{B} = \begin{bmatrix} B_{11} & B_{12} & \cdots & B_{1m} \\ B_{21} & B_{22} & \cdots & B_{2m} \\ \vdots & \vdots & & \vdots \\ B_{m1} & B_{m2} & \cdots & B_{mm} \end{bmatrix},$$

$$\begin{bmatrix} 1 & \mathbf{c_B}^T\mathbf{B}^{-1}\mathbf{A} - \mathbf{c}^T & \mathbf{c_B}^T\mathbf{B}^{-1} \\ \mathbf{0} & \mathbf{B}^{-1}\mathbf{A} & \mathbf{B}^{-1} \end{bmatrix} \begin{bmatrix} Z \\ \mathbf{x} \\ \mathbf{x_s} \end{bmatrix} = \begin{bmatrix} \mathbf{c_B}^T\mathbf{B}^{-1}\mathbf{b} \\ \mathbf{B}^{-1}\mathbf{b} \end{bmatrix} \quad (3)$$

This means that only $\mathbf{B}^{-1}$ needs to be derived to be able to calculate all the numbers used in the simplex method from the original parameters of $\mathbf{A}, \mathbf{b}, \mathbf{c_B}$— providing efficiency and numerical stability.

2.2 Relational Representation

A *relational model* provides a single way to represent data as a two-dimensional table or a *relation*. An n-ary relation being a subset of the Cartesian product of n domains has a collection of rows called *tuples*. Implementing the simplex and revised simplex methods must locate the exact position of the values for the equations and variables of the linear programming problem to be solved. However, the position of the tuples in the table is not relevant in the relational model. Tuple ordering and matrix handling are beyond the standard relational features, and these are the important issues addressed to implement the linear programming solver using the simplex method.

Simplex calculations are most conveniently performed with the help of a series of tables known as *simplex tableaux* (Dantzig, 1963; Hillier and Lieberman, 2001). A simplex tableaux is a table that contains all the information necessary to move from one iteration to another while performing the simplex method. Let $\mathbf{x_B}$ be a column vector of m basic variables obtained by eliminating the nonbasic variables from $\mathbf{x}$ and $\mathbf{x_s}$. Then, the initial tableaux can be expressed as,

$$\begin{bmatrix} Z & 1 & -\mathbf{c}^T & \mathbf{0} & 0 \\ \mathbf{x_B} & \mathbf{0} & \mathbf{A} & \mathbf{I} & \mathbf{b} \end{bmatrix} \quad (4)$$

The algebraic treatment based on the revised simplex method (Hillier and Lieberman, 2001; William

H. Press and Flannery, 2002) derives the values at any iteration of the simplex method as,

$$\begin{bmatrix} Z & 1 & \mathbf{c_B}^T\mathbf{B}^{-1}\mathbf{A}-\mathbf{c}^T & \mathbf{c_B}^T\mathbf{B}^{-1} & \mathbf{c_B}^T\mathbf{B}^{-1}\mathbf{b} \\ \mathbf{x_B} & \mathbf{0} & \mathbf{B}^{-1}\mathbf{A} & \mathbf{B}^{-1} & \mathbf{B}^{-1}\mathbf{b} \end{bmatrix} \quad (5)$$

For the matrices expressed (4) and (5), the first two column elements do not need to be stored into persistent memory. Thus, the simplex tableaux can be a table using the rest of the three column elements in the relational model. Specifically, a linear programming problem in the augmented form (equations (2)) can be seen as a relation:

$$\texttt{tableaux}(\underline{\texttt{id}}, x_1, x_2, \cdots x_n, \texttt{rhs})$$

Every variable of the constraints and the objective function becomes an attribute of the relation, together with the right hand that becomes the `rhs` column on the table. The `id` column serves a *key* that can uniquely determine every variable of the constraints and of the objective function. When the LP problem in augmented form is inputted into the database, every constraint is identified by a unique integer value going from 1 to n in the `id` column, where n is the number of constraints for the problem plus the objective function. Thus by applying relational operations, it is feasible to know the position of every constraint and variable for a linear programming problem, and to proceed with the matrix operations necessary to implement the simplex algorithm.

3 SYSTEM DEVELOPMENT

The availability of real-time databases capable of accepting and solving linear programming problems helps us examine the effectiveness and practical usability in integrating linear programming tools into database environment. Towards this end, a general linear programming solver is developed on top of the *de facto* standard database environment, with the combination of a PHP application for the front-end and a MySQL application for the back-end. Note that the implementation of this linear programming solver is strictly within the database technology, not relying on any outside programming language.

The PHP front-end enables the user to input the number of variables and number of constraints of the linear programming problem to solve. With these values, it generates a dynamic Web interface to accept the values of the objective function and the values of the constraining equations. The Web interface also allows the user to upload a file in a MPS (Mathematical Programming System) format that defines a linear programming problem. The MPS file format serves a standard for describing and archiving linear programming and mixed integer programming problems (Organization, 2007). A special program is built to convert MPS data format into SQL statements for populating an linear programming instance—the main objective of this development is to obtain benchmark performances for large-scale linear programming problems.

The MySQL back-end performs iterative computations by the use of a set of stored procedures precompiled and integrated into the database structure. The system encapsulates an API for processing a simplex method that requires the execution of several SQL queries to produce a solution.

The output of the system is shown in Figure 1, in which each table represents the tableaux containing the values resulted at each iteration. The system presents successive transformations and optimal solution if it exists.

Linear Program Solver

x1	x2	x3	x4	x5	x6	x7	RHS
3	-2	1	1	0	0	0	5
1	3	-4	0	1	0	0	9
0	1	5	0	0	-1	0	1
1	1	1	0	0	0	1	6
-2	3	-1	0	0	0	0	0

x1	x2	x3	x4	x5	x6	x7	RHS
3	0	11	1	0	-2	0	7
1	0	-19	0	1	3	0	6
0	1	5	0	0	-1	0	1
1	0	-4	0	0	1	1	5
-2	0	-16	0	0	3	0	-3

x1	x2	x3	x4	x5	x6	x7	RHS
3.66666666	0	-1.66666666	1	0.66666666	0	0	11
0.33333333	0	-6.33333333	0	0.33333333	1	0	2
0.33333333	1	-1.33333333	0	0.33333333	0	0	3
0.66666667	0	2.33333333	0	-0.33333333	0	1	3
-2.99999999	0	2.99999999	0	-0.99999999	0	0	-9

x1	x2	x3	x4	x5	x6	x7	RHS
4.14285714	0	0	1	0.42857143	0	0.71428571	13.14285714
2.14285717	0	0	0	-0.57142856	1	2.71428572	10.14285717
0.71428572	1	0	0	0.14285714	0	0.57142857	4.71428572
0.28571429	0	1	0	-0.14285714	0	0.42857143	1.28571429
-3.85714286	0	0	0	-0.57142857	0	-1.28571429	-12.85714286

Result: Optimal Solution.

Figure 1: Simplex method iterations and optimal solution.

3.1 Stored Procedure Implementation

Stored procedures can have direct accesses to the data in the database, and run their steps directly and entirely within the database. The complex logic runs inside the database engine, thus faster in processing requests because numerous context switches and a great deal of network traffic can be eliminated. The database system only needs to send the final results back to the user, doing away with the overhead of communicating potentially large amounts of interim

Table 1: Experimental set and measured execution time.

name	*m*	*n*	nonzeros	optimal value	time	standard deviation
ADLITTLE	57	97	465	2.2549496316E+05	1 min. 25 sec.	2.78 sec.
AFIRO	28	32	88	-464.7531428596	35 sec.	1.67 sec.
BLEND	75	83	521	-3.0812149846E+01	1 min. 5 sec.	3.20 sec.
BRANDY	221	249	2150	1.5185098965E+03	2 min. 50 sec.	4.25 sec.

data back and forth (Gulutzan, 2007; Gulutzan and Pelzer, 1999).

Stored procedures are supported by most DBMSs, but there is a fair amount of variation in their syntax and capabilities even their internal effects are almost invisible. Our development uses MySQL version 5.0.22 at the time of this paper writing (as for MySQL version 5, stored procedures are supported). The next code listing is the stored procedure used to create the table to store the linear programming problem to be solved by the application. The first part of the stored procedure consists of the prototype of the function and the declaration of the variables to be used in the procedure.

```
DELIMITER $$
DROP PROCEDURE IF EXISTS
  `lpsolver`.`createTable` $$
CREATE PROCEDURE `lpsolver`.`createTable`
   (constraints INT, variables INT)
BEGIN
  DECLARE i INT;
  DECLARE jiterator VARCHAR(50);
  DECLARE statement VARCHAR(1000);
  DROP TABLE IF EXISTS tableaux;
```

Because of the dynamic nature of the calculations for solving linear programming problems, our stored procedure relies on the extensive use of prepared SQL statements. In the next code block, the SQL statement to create a table is generated on the fly, based on the number of variables and constraints of the problem to solve. The generated procedure is then passed to the database for execution.

```
SET statement = 'CREATE TABLE
    tableaux(id INT(5) PRIMARY KEY, ';
SET i = 1;
table_loop:LOOP
  IF i > constraints + variables + 1 THEN
     LEAVE table_loop;
  END IF;
  SET jiterator = CONCAT('j',i);
  SET statement = CONCAT(statement,
      jiterator);
  SET statement = CONCAT(statement,
      ' DOUBLE DEFAULT 0');
  IF i <= constraints + variables THEN
     SET statement = CONCAT(statement, ', ');
  END IF;
  SET i = i + 1;
  END LOOP table_loop;
  SET statement = CONCAT(statement, ')');
  SET @sql_call = statement;
  PREPARE s1 FROM @sql_call;
  EXECUTE s1;
  DEALLOCATE PREPARE s1;
END $$
DELIMITER ;
```

3.2 Experimental Results

To see the effectiveness of the implementation, various linear programming problems were selected from commonly available Netlib linear programming library (Organization, 2007). As one case, see Table 1 for a sufficiently large problem set. The values m and n indicate the size, $m \times n$, of the coefficient matrix $\mathbf{A}$ in equations (1), or equivalently, m is the number of constraints and n is the number of decision variables.

All experiments were performed by an Intel 586 based standalone machine with 1.2 GHz CPU and 512 MB memory that running MySQL 5.0.22. The data values were extracted from Netlib MPS files to populate the problems into the database prior to run the simplex method. The time measured does not include this data preparation process, but only the execution of the stored procedure to produce a solution. The time listed in the following table is the average of ten executions of each problem. The set of results shown in Table 1 are based on the implementation of the revised simplex method contained in the stored procedures.

One limiting factor is the fact that MySQL allows to have up to 1000 columns on a table. Given that our implementation is based on mapping of a simplex tableaux into a database relation, the number of variables plus the number of constraints cannot exceed the number of columns allowed for a MySQL table. This prohibited us from testing the problems in the Netlib library that exceed the column size of 1000. Finally, we observed one problem when trying to find optimal solutions for larger problems with higher numbers of columns, variables and zero elements. The computation never came to an end, indicating that the problem had become unbounded, which can be attributed to the tableaux becoming *ill-conditioned* as a consequence of truncation errors resulted from repeated matrix operations.

4 RELATED WORK

A vast amount of effort for the establishment of theory and practice is observed today. Certain special cases of linear programming, such as network flow problems and multi-commodity flow problems are considered important enough to have generated much research on specialized algorithms for their solution (Winston, 1994; Thomas H. Cormen and Stein, 2001; Hillier and Lieberman, 2001). A number of algorithms for other types of optimization problems work by solving linear programming problems as subproblems. Historically, ideas from linear programming have inspired many of the central concepts of optimization theory, such as duality, decomposition, and the importance of convexity and its generalizations (Hillier and Lieberman, 2001).

There are approaches considered to fit a linear programming model, such as integer programming and nonlinear programming (Alexander, 1998; Richard, 1991; Hillier and Lieberman, 2001). But, our research focuses on the area of iterative methods for solving linear systems. Some of the most significant contributions and the chain of contributions building on each other are summarized in (Saad and van der Vorst, 2000), especially a survey of the transition from simplex methods to interior-point methods is presented in (Wang, 99). In terms of implementation techniques, the work of (Morgan, 1976; Shamir, 1987) provided us with introductory sources for reference. There are online materials such as (Optimization Technology Center and Laboratory, 2007; Organization, 2007) to help us understand the details and plan for experimental design.

5 CONCLUSION

The subject of this research is to respond a lack of database tools for solving a linear programming problem defined within a database.

We described the aim and approach for integrating a linear programing method into today's database system, with our goal in mind to establish a seamless and transparent interface between them. As demonstrated, this is feasible by the use of stored procedures, the emerging database programming standard that allows for complex logic to be embedded as an API in the database, thus simplifying data management and enhancing overall performance.

We implemented a general linear programming solver on top of the PHP and MySQL software layers. The experiments with several benchmark problems extracted from the Netlib library showed its correct optimal solutions and basic performance measures.

The contents of this paper are the first dissemination of our upcoming series of publications—we plan to extend this research into several directions. First, the limiting factor of MySQL's table column size for the implementation needs to be addressed. Also, the code must be optimized to reduce the overall execution time. Second, more experiments must be done to collect additional performance measures. Furthermore, other commercial databases such as Oracle should be included for comparative study, and nonlinear and other optimization methods should also be explored.

REFERENCES

Alexander, S. (1998). *Theory of Linear and Integer Programming*. John Wiley & Sons, New York, NY.

Dantzig, G. B. (1963). *Linear Programming and Extensions*. Princeton University Press, Princeton, N.J.

Gulutzan, P. (2007). *MySQL 5.0 New Features: Stored Procedures*. MySQL AB, http://www.mysql.com.

Gulutzan, P. and Pelzer, T. (1999). *SQL-99 Complete, Really*. CMP Books.

Hillier, F. S. and Lieberman, G. J. (2001). *Introduction to Operations Research*. McGraw-Hill, 8th edition.

Karmarkar, N. K. (1984). A new polynomial-time algorithm for linear programming and extensions. *Combinatorica*, 4:373–395.

Morgan, S. S. (1976). A comparison of simplex method algorithms. Master's thesis, University of Florida.

Optimization Technology Center, N. U. and Laboratory, A. N. (2007). The linear programming frequently asked questions.

Organization, T. N. (2007). The netlib repository at utk and ornl.

Richard, B. D. (1991). *Introduction To Linear Programming: Applications and Extensions*. Marcel Dekker, New York, NY.

Saad, Y. and van der Vorst, H. (2000). Iterative solution of linear systems in the 20-th century. *JCAM*.

Shamir, R. (1987). The efficiency of the simplex method: a survey. *Manage. Sci.*, 33(3):301–334.

Thomas H. Cormen, Charles E. Leiserson, R. L. R. and Stein, C. (2001). *Introduction to Algorithms, Chapter29: Linear Programming*. MIT Press and McGraw-Hill, 2nd edition.

Walsh, G. R. (1985). *An Introduction to Linear Programming*. John Wiley & Sons, New York, NY.

Wang, X. (99). From simplex methods to interior-point methods: A brief survey on linear programming algorithms.

William H. Press, Saul A. Teukolsky, W. T. V. and Flannery, B. P. (2002). *Numerical Recipes in C++: The Art of Scientific Computing*. Cambridge University.

Winston, W. L. (1994). *Operations Research, Applications and Algorithms*. Duxbury Press.

A DISTRIBUTED REINFORCEMENT LEARNING CONTROL ARCHITECTURE FOR MULTI-LINK ROBOTS
Experimental Validation

Jose Antonio Martin H.
Dep. Sistemas Informáticos y Computación, Universidad Complutense de Madrid
C. Prof. José García Santesmases, 28040, Cuidad Univesitaria, Madrid, Spain
Email: jamartinh@fdi.ucm.es

Javier de Lope
Dept. Applied Intelligent Systems, Universidad Politécnica de Madrid
Campus Sur, Ctra. Valencia, km. 7, 28031 Madrid, Spain
Email: javier.delope@upm.es

Keywords: Reinforcement Learning, Multi-link Robots, Multi-Agent systems.

Abstract: A distributed approach to Reinforcement Learning (RL) in multi-link robot control tasks is presented. One of the main drawbacks of classical RL is the combinatorial explosion when multiple states variables and multiple actuators are needed to optimally control a complex agent in a dynamical environment. In this paper we present an approach to avoid this drawback based on a distributed RL architecture. The experimental results in learning a control policy for diverse kind of multi-link robotic models clearly shows that it is not necessary that each individual RL-agent perceives the complete state space in order to learn a good global policy but only a reduced state space directly related to its own environmental experience. The proposed architecture combined with the use of continuous reward functions results of an impressive improvement of the learning speed making tractable some learning problems in which a classical RL with discrete rewards (-1,0,1) does not work.

1 INTRODUCTION

Reinforcement Learning (RL) (Sutton and Barto, 1998) is a paradigm of Machine Learning (ML) that consists in guiding the learning process by rewards and punishments.

One of the most important breakthroughs in RL was the development of Q-Learning (Watkins, 1989). In a simple form its learning rule is:

$$Q(s,a) = Q(s,a) + \alpha[r + \gamma \max_a Q(s_{t+1},*) - Q(s,a)] \quad (1)$$

Note that this is the classical *moyenne adaptive modifiée* technique (Venturini, 1994) which adaptively approximates the average μ of a set $x = (x_1 ... x_\infty)$ of observations:

$$\mu = \mu + \alpha[x_t - \mu], \quad (2)$$

thus replacing the factor x_t in (2) with the current reward r plus a fraction $\gamma \in (0,1]$ of the best possible approximation to the maximum future expected reward $\max_a Q(s_{t+1},*)$:

$$\mu = \mu + \alpha[r + \gamma \max_a Q(s_{t+1},*) - \mu], \quad (3)$$

we are indeed approximating the average of the maximum future reward when taking the action a at the state s.

In (Watkins and Dayan, 1992) the authors proved convergence of Q-Learning to an optimal control policy with probability **1** under certain single assumptions see (Sutton and Barto, 1998, p.148).

In this paper we present a distributed approach to RL in robot control tasks. One of the main drawbacks of classical RL is the combinatorial explosion when multiple states variables and multiple actuators are needed to optimally control a complex agent in a complex dynamical environment. As an example, a traditional RL system for the control of a robot with multiple actuators (i.e. multi-link robots, snake-like robots, industrial manipulators, etc.) when the control task relies on feedback from the actuators. This situation produces a huge table of $states \times actions$ pairs which, in the best case, assuming enough memory to store the table, the learning algorithm needs a huge amount of time to converge, indeed the problem could become intractable.

Also, for some nonlinear parameterized function approximators, any temporal-difference (TD) learning method (including Q-learning and SARSA) can become unstable (parameters and estimates going to infinity) (Tsitsiklis and Roy, 1996). For the most part, the theoretical foundation that RL adopts from dynamic programming is no longer valid in the case of function approximation (Sutton and Barto, 1998). For these reasons the development of distributed algorithms relying on partial observations of states and individual action selections mechanisms becomes crucial to solve such intractable problems.

Our approach avoids this drawback based on a distributed architecture. The work is based on the hypothesis that such complex agents can be modeled as a colony of single interacting agents with single state spaces as well as single action spaces. This hypothesis is verified empirically by the means of clear experimental results showing the viability of the presented approach.

The rest of the paper is organized as follows. First we introduce some previous works on RL for robot controlling. In the next section we present the basis of our distributed RL control approach, including a pseudo-code of the proposed algorithm. Next we define an application framework in which we have carried out the experiments. For the experiments we use two multi-link robot models, a three-link planar robot and an industrial SCARA.

2 RL IN ROBOT CONTROL

RL has been applied successfully to some robot control task problems (Franklin, 1988; Lin, 1993; Mataric, 1997; Rubo et al., 2000; Yamada et al., 1997; Kretchmar, 2000; Kalmar et al., 2000; El-Fakdi et al., 2005) and is one of the most promising paradigms in robot learning. This is mainly due to its simplicity. The main advantages of RL over other methods in robot learning and control are:

1. There is no need to know a prior model of the environment. This is a crucial advantage because in most complex tasks a model of the environment is unknown or is too complex to describe.
2. There is no need to know previously what actions for each situation must be presented to the learner.
3. The learning process is online by directly interacting with the environment.
4. It is capable of learning from scratch.

RL can be applied to robot control tasks in a very easy way. The basic elements of a RL learning system are:

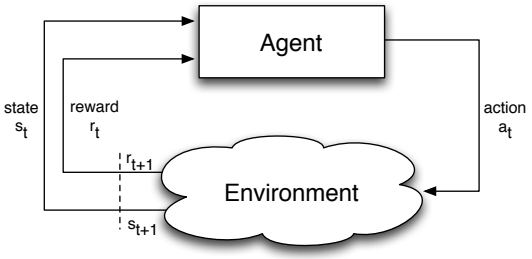


Figure 1: The basic model of an RL system.

1. An agent or a group of agents (i.e. mobile robots) that perceives the environment (typically called state S_t) and behaves on the environment producing actions (a_t).
2. The environment in which the agents live, that could be a simulated world or the real world.
3. A reward signal r_t that represents the evaluation of the state and it is used by the agents to evaluate its behavioral policies.

Fig. 1 shows the agent-environment interaction in RL. The control cycle can be used directly to control any robotic task where the agent represents a robot.

The majority of RL algorithms rely on the condition that the state signal S_t fulfills the Markov property that is, the signal S_t must contain for the time step (t) all the information needed to take an optimal action at time step (t). In other words we can define that a signal state S_t fulfills the Markov property if and only if there is no information that could be added into S_t that could produce a more optimal action than the original signal control.

We use SARSA as the basis for our RL algorithm. SARSA has been proved to converge under the same set of single assumptions that Q-Learning (Sutton and Barto, 1998) but empirically it outperforms the Q-Learning's performance. The pseudo-code of the SARSA algorithm is described in Fig. 2 which uses a slight variation of the learning rule shown in (1).

3 PROPOSED ARCHITECTURE

In this work we propose a distributed architecture for RL in order to prevent the combinatorial explosion when multiple states and actuators are needed. The main feature of this architecture is the division of the learning problem into different agents each one for every different actuator (i.e. joint or motor). Fig. 3 shows the basic control cycle of this architecture.

Initialize Q arbitrarily
Repeat(for each episode)
  Observe s
  Select a for s by ε-greedy policy
  Repeat(for n steps)
    Take action a, observe r,s'
    Select a' for s' by ε-greedy policy
    $Q(s,a) = Q(s,a) + \alpha[r + \gamma Q(s',a') - Q(s,a)]$
  $s = s'$;$a = a'$
until s is terminal

Figure 2: SARSA: RL control algorithm.

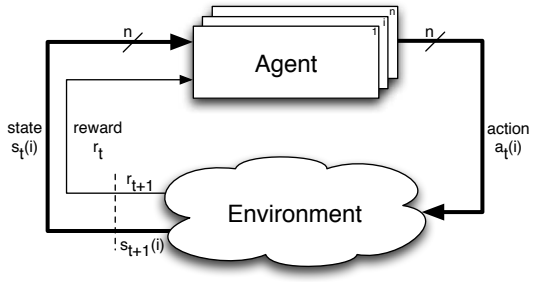


Figure 3: The basic model of the Distributed Approach to RL.

In Fig.3 we can see that the RL signal r_t is the same for all agents and the state signal s_{t+1} is propagated against all agents despite each individual agent can filter it and use only the information needed to behave properly. Also, we must note that each agent emits a different action to the environment. Thus, each agent will have its own independent memory in which to store its respective knowledge and hence the quantity of state-action pairs in the global system is considerably less than within a full state-action space due to the additive scheme instead of the multiplicative scheme which is the main cause of the curse of dimensionality. Fig. 4 shows the proposed algorithm based on the SARSA algorithm which reflects the proposed multi-agent control architecture.

Within this distributed approach every agent will have its own Q-table Q_i and its own perceptions $f_i(s_i)$. In the case of multi-link robots, each agent will perceive as its own state signal only the current state of its respective actuator plus the information of the goal reaching index.

3.1 Continuous Rewards

Traditionally RL systems work with discrete rewards (i.e. [-1, 0, 1]) but this is not mandatory due to the algorithms actually can handle scalar reward signals.

Initialize Q_i arbitrarily ; $\forall i = 1,...,n$
Repeat(for each episode)
  Observe $(s_1,...,s_n)$
  Select a_i for s_i by ε-greedy policy ; $\forall i = 1,...,n$
  Repeat(for J steps)
    Take actions $a_1,...,a_n$, observe r, $s'_1,...,s'_n$
    Select a'_i for s'_i by ε-greedy policy ; $\forall i = 1,...,n$
    $Q_i(s_i,a_i) = Q_i(s_i,a_i) + \alpha[r + \gamma Q(s'_i,a'_i) - Q(s_i,a_i)]$
  $s_i = s'_i$;$a_i = a'_i$
until s is terminal

Figure 4: MA-SARSA: A Multi-agent RL control algorithm.

Indeed the conditions for proper convergence assume scalar rewards signals. In our model we use continuous rewards functions as a way to improve convergence and learning speed. The use of continuous rewards signals may lead to some local optima traps but the exploratory nature of RL algorithms tends to escape from these traps.

One elegant way to formulate a continuous reward signal in the task of reaching a goal is simply to use the inverse Euclidean distance plus 1 as shown in (4):

$$reward = \frac{\beta}{1+d^n}, \tag{4}$$

where β is a scalar that controls the absolute magnitude of the reward, d is the Euclidean distance from the goal position and n is an exponent which is used to change the form in which the function responds to changes in the Euclidean distance. In out model we use generally (4) as the reward when the state is terminal and the goal is reached otherwise we use another continuous function as a penalty function for every action that does not reach the goal as shown in (5).

$$punishment = -\beta d^n - \varepsilon, \tag{5}$$

where the term ε is used to include some constraints that the system engineer considers appropriate.

We have used this same kind of continuous rewards functions in past works with successful results for multi-objective RL systems (Martin-H. and De-Lope, 2006).

4 EXPERIMENTAL APPLICATION FRAMEWORK

Experiments have been realized on two multi-link robot models: a three-link planar robot and a SCARA robot. The experiments are made with the intention that the robots learn their inverse kinematics in order

to reach a desired configuration in the joint and Cartesian spaces. For this reason we have selected both types of manipulators.

4.1 Three-Link Planar Robot

The three link planar robot is a configuration which is generally studied by the Robotics community and its kinematics is well-known. It includes some redundancy in the calculus of the inverse kinematics by means of analytic procedures. For this reason, it is very interesting for the verification of our methods in this kind of mechanical structures. As we have commented previously the same approach could be applied for controlling the movement and body coordination of snake-like robots.

Fig. 5 shows a diagram of the robot model [1]. Its Denavit-Hartenberg parameters are described in the Table 1 and its direct kinematic equation is defined in (6).

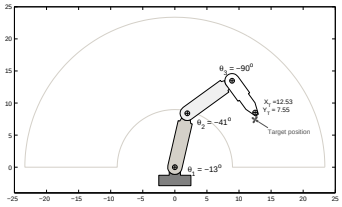


Figure 5: Three link-planar robot model.

Table 1: Three link planar robot D-H parameters.

i	θ_i	d_i	a_i	α_i
1	θ_1	0	$a_1 = 8.625$	0
2	θ_2	0	$a_2 = 8.625$	0
3	θ_3	0	$a_3 = 6.125$	0

$$T^3 = \begin{pmatrix} C_{123} & -S_{123} & 0 & a_1C_1 + a_2C_{12} + a_3C_{123} \\ S_{123} & C_{123} & 0 & a_1S_1 + a_2S_{12} + a_3S_{123} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad (6)$$

where S_{123} and C_{123} correspond to $\sin(\theta_1 + \theta_2 + \theta_3)$ and $\cos(\theta_1 + \theta_2 + \theta_3)$, respectively (and equally for S_1, S_{12}, C_1 and C_{12}), θ_1, θ_2 and θ_3 define the robot joint coordinates —θ_1 for the first joint, located near to the robot base, θ_2 for the middle joint and θ_3 for the joint situated in the final extreme— and a_1, a_2 and a_3 correspond to the physical length of every link, first, second and third, respectively, numbering from the robot base.

[1] The original model is by Matt Kontz from Walla Walla College, Edward F. Cross School of Engineering, February 2001.

4.2 Scara Robot

The SCARA robot was selected due to it is a widely used robot in the industry. It has well-known properties for the use as manufacturing cells assisting conveyor belts, electronic equipment composition, welding tasks, and so on. In this case we are using a real three dimension robot manipulator so the agents have to try to reach an objective configuration in a 3D space being able to generate an approaching real-time trajectory when the agent system is completely trained.

A physical model of the SCARA robot is shown in Fig. 6. The Denavit-Hartenberg parameters are defined in Table 2 and the direct kinematic matrix is shown in (7).

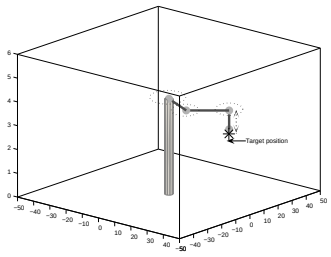


Figure 6: SCARA robot model.

Table 2: SCARA robot D-H parameters.

i	θ_i	d_i	a_i	α_i
1	θ_1	0	$a_1 = 20$	0
2	θ_2	0	$a_2 = 20$	π
3	θ_3	0	0	0
4	0	d_4	0	0

$$T^4 = \begin{pmatrix} C_{12-3} & -S_{12-3} & 0 & a_1C_1 + a_2C_{12} \\ S_{12-3} & C_{12-3} & 0 & a_1S_1 + a_2S_{12} \\ 0 & 0 & -1 & -d_4 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad (7)$$

where S_{12-3} and C_{12-3} correspond to $\sin(\theta_1 + \theta_2 - \theta_3)$ and $\cos(\theta_1 + \theta_2 - \theta_3)$, respectively (and equally for S_1, S_{12}, C_1 and C_{12}), θ_1, θ_2, θ_3 and d_4 are the joint parameters for the shoulder, elbow, wrist and prismatic joints, respectively, and a_1 and a_2 the lengths of the arm links.

4.3 Experiments Design

The control task consists of to reach a continuously changing goal position of the robot end-effector by means of a random procedure. The goal positions are defined in such a way that they are always reachable for the robot. Thus, the learning process needs to develop an internal model of the inverse kinematics of the robot which is not directly injected by the designer in a *aprioristic* way. Through the different trials, a model of the robot inverse kinematics is *learned* by the system.

When a goal position is generated, the robot tries to reach it. Each trial can finish as a success episode, i.e. the robot reaches the target at a previously determined time or as a fail episode, i.e. the robot is not able to adopt a configuration to reach the goal position. In both cases the system parameters are updated using the previously defined method and a new goal position is randomly generated.

Once the systems parameters have been tuned by the RL process, the system has learned an inverse kinematics model of the robot. Such a model can be used for real time control of the robot. The learning process duration depends on the initial conditions and the randomly generated positions since RL relies on a kind of stochastic search, but as it is shown in the experimental results this time is very short. We will comment in detail on the results in the next section.

5 EXPERIMENTAL RESULTS

The test procedure defined above has been applied to both robots the three-link planar robot and the SCARA robot. Although the results are very similar we offer separate results for each robot.

5.1 The Three-link Planar Robot

The three-link planar robot has a relatively simple inverse kinematics, although its three rotation axes are in parallel, generating multiple candidate configurations for the same position and orientation. The learning system produces extremely good results, finding a model of the inverse kinematics in 50 episodes in average, being the best case of 4 episodes. The average number of episodes is obtained by a sequence of 30 multiple restarting experiments and calculating the point at which the convergence is achieved for each run.

Fig. 7 shows the learning curve for this best case. As we can observe, before the system runs 4 episodes, the controller converges to the final solution. The peaks displayed are due to the randomly nature of the experiment: the vertical axis represents the steps that were needed to reach the goal, if the new randomly generated goal is far from the current position, then more steps will be needed in order to reach the goal position.

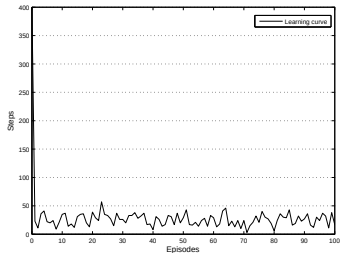


Figure 7: Learning curve for the three-link planar robot model for 100 episodes.

Fig. 8 shows a trace of the behavior of the three-link planar robot for four consecutive goals, It can be seen that there is some little noise in the trajectory in some segments of the paths between consecutive goals. This noise is the product of the nature of the goal reaching information due to some zero-crossings at the orthogonal axes (x,y), this must be interpreted as search for accuracy in the goal reaching attempt. We must recall that we are using a mixture of memory based approach and a reactive mechanism thus these oscillations are a natural consequence of a reactive behavior. This noise can be reduced by some methods to prevent oscillations in which we are working as a future improvement.

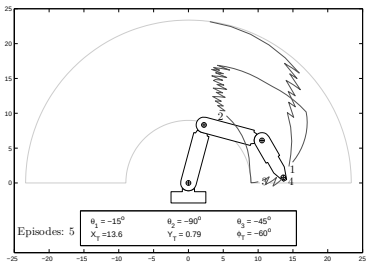


Figure 8: Trace for the three-link planar robot for four consecutive goals.

5.2 The Scara Robot

The SCARA robot presents very similar results. Fig. 9 shows a learning curve for 100 episodes. In these cases the learning process converges slower than the one of previous robot. The abrupt peaks that appears in the first episodes show the adaptation pro-

cess (learning). For the SCARA robot 67 episodes are needed as average to find a solution, obtaining a best case after 23 episodes.

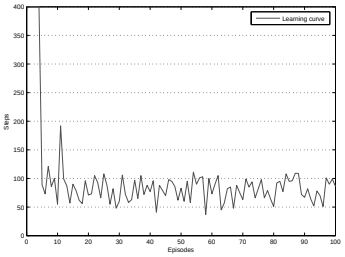


Figure 9: Learning curve for the SCARA robot model for 100 episodes.

Fig. 10 shows a trace of the behavior of the SCARA robot for four consecutive goals, in this picture the same oscillations can be seen as in the trace of the three-link planar robot due to the zero-crossing effect explained above.

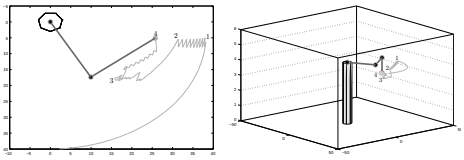


Figure 10: Trace for the SCARA robot for four consecutive goals.

6 CONCLUSIONS

A distributed approach to RL in robot control tasks has been presented. To verify the method we have defined an experimental framework and we have tested it on two well-known robotic manipulators: a three-link planar robot and an industrial SCARA.

The experimental results in learning a control policy for diverse kind of multi-link robotic models clearly shows that it is not necessary that the individual agents perceive the complete state space in order to learn a good global policy but only a reduced state space directly related to its own environmental experience. Also we have shown that the proposed architecture combined with the use of continuous reward functions results in an impressive improvement of the learning speed making tractable some learning problems in which a classical RL with discrete rewards $(-1,0,1)$ does not work. Also we want to adapt the proposed method to snake-like robots. The main drawback in this case could be the absence of a base which fixes the robot to the environment and its implication in the definition of goal positions.

ACKNOWLEDGEMENTS

This work has been partially funded by the Spanish Ministry of Science and Technology, project DPI2006-15346-C03-02.

REFERENCES

El-Fakdi, A., Carreras, M., and Ridao, P. (2005). Direct gradient-based reinforcement learning for robot behavior learning. In *ICINCO 2005*, pages 225–231. INSTICC Press.

Franklin, J. A. (1988). Refinement of robot motor skills through reinforcement learning. In *Proc. of* 27^{th} *Conf. on Decision and Control*, pages 1096–1101, Austin, Texas.

Kalmar, Szepesvari, and Lorincz (2000). Modular reinforcement learning: A case study in a robot domain. *ACTACYB: Acta Cybernetica*, 14.

Kretchmar, R. M. (2000). *A synthesis of reinforcement learning and robust control theory*. PhD thesis, Colorado State University.

Lin, L.-J. (1993). Scaling up reinforcement learning for robot control. *Machine Learning*.

Martin-H., J. A. and De-Lope, J. (2006). Dynamic goal coordination in physical agents. In *ICINCO 2006*, pages 154–159. INSTICC Press.

Mataric, M. J. (1997). Reinforcement learning in the multi-robot domain. *Auton. Robots*, 4(1):73–83.

Rubo, Z., Yu, S., Xingoe, W., Guangmin, Y., and Guochang, G. (2000). Research on reinforcement learning of the intelligent robot based on self-adaptive quantization. In *Proc. of the 3rd World Congr. on Intelligent Control and Automation. IEEE, Piscataway, NJ, USA*, volume 2, pages 1226–9.

Sutton, R. S. and Barto, A. G. (1998). *Reinforcement Learning, An Introduction*. The MIT press.

Tsitsiklis, J. N. and Roy, B. V. (1996). Analysis of temporal-diffference learning with function approximation. In Mozer, M., Jordan, M. I., and Petsche, T., editors, *NIPS*, pages 1075–1081. MIT Press.

Venturini, G. (1994). Adaptation in dynamic environments through a minimal probability of exploration. In *SAB94*, pages 371–379, Cambridge, MA, USA. MIT Press.

Watkins, C. J. (1989). *Models of Delayed Reinforcement Learning*. PhD thesis, Psychology Department, Cambridge University, Cambridge, United Kingdom.

Watkins, C. J. and Dayan, P. (1992). Technical note Q-learning. *Machine Learning*, 8:279.

Yamada, S., Watanabe, A., and Nakashima, M. (1997). Hybrid reinforcement learning and its application to biped robot control. In *NIPS*. The MIT Press.

ACTIVE LEARNING IN REGRESSION, WITH APPLICATION TO STOCHASTIC DYNAMIC PROGRAMMING

Olivier Teytaud*, Sylvain Gelly* and Jérémie Mary* **
*TAO (Inria, Univ. Paris-Sud, UMR CNRS-8623), France
**Grappa (Inria Univ. Lille), France
teytaud@lri.fr, gelly@lri.fr,mary@lri.fr

Keywords: Intelligent Control Systems and Optimization, Machine learning in control applications, Active learning.

Abstract: We study active learning as a derandomized form of sampling. We show that full derandomization is not suitable in a robust framework, propose partially derandomized samplings, and develop new active learning methods (i) in which expert knowledge is easy to integrate (ii) with a parameter for the exploration/exploitation dilemma (iii) less randomized than the full-random sampling (yet also not deterministic). Experiments are performed in the case of regression for value-function learning on a continuous domain. Our main results are (i) efficient partially derandomized point sets (ii) moderate-derandomization theorems (iii) experimental evidence of the importance of the frontier (iv) a new regression-specific user-friendly sampling tool less-robust than blind samplers but that sometimes works very efficiently in large dimensions. All experiments can be reproduced by downloading the source code and running the provided command line.

1 INTRODUCTION

As pointed out in e.g. (Cohn et al., 1995a), the ability of the learner to select examples and modify its environment in order to get better examples is one of the main point in learning. In this model of learning, the learning algorithm is typically made of (i) a sampler, that chooses points in the domain, and (ii) a passive learner that takes these points and their labels as provided by some unknown oracle (target concept in classification or target function in regression). Various forms of active learning have been proposed:

1. Blind approaches in which points to be labelled by the oracle are well distributed in the domain, e.g. in a quasi-random manner, without taking into account the labels (Cervellera and Muselli, 2003); in this case, the process can be splitted in 3 successive steps: (i) sample the points, (ii) label by the oracle, (iii) learn the target concept/function;

2. Non-blind approaches, in which the sampler uses the labels provided by the oracle in order to choose the next points to be labelled; possibly, the learner is embedded in the sampler. These approaches use various criteria; (Lewis and Gale, 1994) chooses examples with the maximum uncertainty of the learner, (Seung et al., 1992) chooses examples that reduce maximally the size of the version space. Other approaches include (Schohn and Cohn, 2000), with application to Support Vector Machines (SVM), (Cohn et al., 1995b), with application to Neural Nets.

Limitations of non-blind approaches. In spite of the fact that the second approach is much more general, the superiority of non-blind approaches is often unclear, due to the nice robustness properties of blind approaches. If you exploit properties of the problem to avoid sampling areas that are probably less interesting, you can miss some ”surprisingly” interesting areas. Pessimistic theorems on the possibility of active learning can be found in e.g. (Vidyasagar, 1997), in a worst-case scenario. Mainly, as a conclusion of the state of the art, the best applied results for non-blind active-learning concern moderately large families of functions; classification more than regression, decision trees more than neural networks or SVM. In particular, we think that a main advantage of viability approaches in reinforcement learning is that it reduces the problem (with loss of generality unfortunately) to a classification problem in which active learning is more stable (Chapel and Deffuant, 2006). This is not very surprising as for example, choosing

points close to boundaries, what makes sense in classification or with regression-trees but not in numeric regression, is a good solution for efficient active learning. We propose in this paper a new non-blind approach, not always better than blind approaches as we will see in the sequel, but that (i) is easy to use for any type of state space (continuous or not) (ii) can be parametrized continuously from the pure blind approach to a very deterministic sampling (iii) can easily integrate expert information about the sampling.

Limitations of deterministic approaches. Deterministic samplings are not always better than pure random sampling. E.g. in numerical integration, deterministic blind samples are much better than random points for various criteria in small dimensions, but when the dimension increases, these strictly deterministic approaches (Sloan and Woźniakowski, 1998) have strong limitations (see also (Kearns et al., 1999; Rust, 1997) for some interesting properties of random sampling in the specific case of control problems). Best results in the traditional field of quasi-random sequences, namely integration, now come from *randomized* quasi-random sequences, in particular when dimension increases (L'Ecuyer and Lemieux, 2002), whereas former best sequences were strictly deterministic ones. This is a somewhat surprising element in the history of quasi-random sequences. We show here both empirically and theoretically a similar superiority of randomized, yet non-naive, samplings in the case of active learning. We also conclude empirically to some related limitations of non-blind approaches w.r.t blind approaches in terms of robustness but our theorems only concern almost-deterministic samplings, and non-blind approaches are concerned only if too strongly deterministic.

Why active learning is very important in dynamic problems. The importance of the exploration step is particularly strong in reinforcement learning, where exploitation (learning) is deeply mixed with exploration (gathering information); this is why active learning is decisive in particular for dynamic problems. A main trouble, with respect to elements above, is that value functions used in reinforcement learning lead to regression problems and not classification ones. Note that many works about active sampling of the environment use simulations; see e.g. the pioneer work (Barto et al., 1993) and many subsequent works. We will here avoid these techniques, that have strong advantages but also limitations as they can miss interesting parts of the state space that are not seen in simulations due to poor initial policies. We here only consider sampling methods that sample as efficiently as possible the 'full' domain. This is orthogonal to, and not competing with, simulation-based samplers. Examples of non simulation-based active learning for dynamic problems have been provided in (Munos and Moore, 1999) (active discretization of the domain), (Chapel and Deffuant, 2006) (active learning for SVM in a viability-framework which reformulates the regression task in a classification task).

Overview of results. We will here study the following questions, in the case of regression:
1. Is non-blind active-learning better than blind active learning? Essentially, we will see significant differences in both senses; for some problems the results are indeed much worse. This is an experimental answer but we provide also proofs for the related robustness of randomized techniques w.r.t too-much-deterministic methods. A non-blind sampler termed EAS is defined and tested.
2. Are deterministic blind-samplers better than random ones? Essentially, we prove (th. 2.4) robustness (universal consistency) results for random samplers, and non robustness for deterministic samplers. We then show that the quantity of randomness can be strongly reduced, in a classical random-shift framework preserving both (i) improved convergence rates (as shown in (Cervellera and Muselli, 2003)) for "smooth-enough" target-functions and (ii) universal consistency (as shown in this paper) for any target function. We propose and test an efficient blind-sampler termed GLD.

2 MATHEMATICAL ANALYSIS

We study here the derandomized samplings, i.e. sampling with less randomness. It has been shown (Cervellera and Muselli, 2003) that derandomizing improves convergence rates for smooth target functions when compared to naive random samplers. We show that however, some robustness properties require a random part (theorem 2.3), but that this random part can be reduced to combine (i) the robustness of naive random sampling (ii) the improved convergence rates of deterministic samplers for "easy" cases.

Definition 2.1 *Consider a domain $D = [0,1]^d$. Note E^* the set of finite families of elements of E. A* **learner** *A on D is a computable mapping from $(D \times \mathbb{R})^*$ to $\mathbb{R}^D$. Let $\mathcal{A}$ the set of learners. A* **sampler** *S on D is a computable mapping from $(D \times \mathbb{R})^* \times \mathcal{A}$ to D. If S can be restricted to a computable mapping from $(D \times \mathbb{R})^*$ to D (i.e. if the $\mathcal{A}$-component is not used) then it is called a* **blind-sampler** *A* **sample-learner** *on D, based on the sampler S and on a learner A, and noted $S+A$, is an algorithm that takes as input*

an effective method f on a domain D and that can be written as follows:

1. *set $n \leftarrow 0$;*
2. *choose x_n in D defined by $x_n \leftarrow S((x_0,y_0)\ldots,(x_{n-1},y_{n-1}),A)$;*
3. *define $y_n \leftarrow f(x_n)$;*
4. *set $f_n = A((x_0,y_0),\ldots,(x_n,y_n))$.*
5. *set $n \leftarrow n+1$ and go back to step 2.*

A sampler is said **almost-deterministic** *(AD) if for some $n \mapsto k(n)$ finite it only uses $k(n)$ random bits before choosing x_n (i.e. before the n^{th} epoch of the algorithm above). A learner is said* **almost-deterministic** *(AD) if for each run it only uses a finite number of random bits that only depends on the length of its inputs. A sample-learner on D is said* **universally consistent** *(UC) if, for any measurable f with values in $[0,1]$, almost surely $\int_D (f_n(x)-f(x))^2 dx \to 0$ as $n \to \infty$. We say that a sampler S on D is* **universally consistent** *(UC) if for at least one AD learner A, the sampler-learner $S+A$ is UC.*

We require in the definition of UC for a sampler that the learning algorithm is AD because otherwise stochasticity of the learner can be used for randomizing the sampler. Therefore, to distinguish AD-samplers and non-AD-samplers, we have to add this restriction.

Theorem 2.2 *The random sampling is UC.*

Proof: By any UC-algorithm (see e.g. (Devroye et al., 1994)).

Theorem 2.3 (UC samplers are stochastic) *Consider S an AD-sampler. Then, for any AD-learner A, $S+A$ is not UC. Therefore, S is not UC.*

Proof: Consider S and A, respectively an AD-sampler and an AD-learner. Then, consider $x_0,\ldots,x_n,\ldots$ the sequence of points provided by the sampler if f is the constant function equal to 1 (this sequence might be stochastic or not). By definition of AD, x_i takes values in a finite domain of cardinal c_i. Therefore, all the x_n take values in some countable domain V. Consider now g_p, for $p \in [0,1[$, the function equal to 1 in V, and to p in $D \setminus V$. If $S+A$ is UC, then on a target $f = g_p$ almost surely f_n converges in norm L^2 to the constant function equal to p. However, for all $f = g_p, p \in [0,1[$, f_n is distributed in the same manner, as f_n depends only on the f values in V. With $p = 0$ and $p = \frac{1}{2}$ for example, this leads to a contradiction. $S+A$ can not be UC. □

We showed in theorem 2.2 that random sampling is UC and in theorem 2.3 that no AD-sampling can be UC. But where is the limit? We show in the following theorem that a moderate derandomization, yet using a continuous random seed, can lead to universal consistency. Note that the result includes quasi-random sequences with a simple random shift, and therefore includes samplings that have proved faster under some smoothness hypothesis than the random sampling (Cervellera and Muselli, 2003).

Note that the theorem below has a very moderate requirement in terms of discrepancy (only convergence to 0, whereas low-discrepancy sequences typically verify $O(\log(n)^d/n)$).

Theorem 2.4 (A random shift is enough) *Consider a sampler S that outputs $x_0,\ldots,x_n$ and defined as follows:*

1. randomly uniformly draw $s \in D$.

2. $y_0,\ldots,y_n,\ldots$ is a deterministic sequence in D with dispersion

$$\sup_{x \in D} \inf_{i \in [[1,n]]} ||x - y_i||_\infty = O(1/n^{1/d})$$

and discrepancy

$$\sup_{r \in [0,1]^d} |\frac{1}{n} \#\{i \in [[1,n]]; \forall j, x_{i,j} \leq r_j\} - \pi_{i \in [[1,d]]} r_i| \to 0$$

as $n \to \infty$ where $x_{i,j}$ is the j^{th} component of x_i.

3. for any n, $x_n = y_n + s$ modulo 1 (i.e. x_n is such that $y_n - x_n \in \mathbb{Z}^d$ and $x_n \in [0,1[^d$).

Then, S is UC.

Interpretation: Adding a uniform random shift to a deterministic sequence is enough to get a UC sampler with the same improved convergence rates as in (Cervellera and Muselli, 2003) for smooth target functions. Comparing theorems 2.3 and 2.4 show that randomness is required for UC, but the quantity of randomness can be much smaller than in pure random sampling.

Proof: See `http://www.lri.fr/~teytaud/ldsfordplong.pdf`.

3 SAMPLING METHODS (SM) AND APPLICATIONS

We present below (i) the problem of function value learning (ii) some blind point-sets (iii) a non-blind learner-independent sampler (iv) our experiments on the OpenDP set of benchmark-problems.

3.1 Active Function Value Learning

We introduce reinforcement learning and stochastic dynamic programing (SDP) in a very short manner; see (Bertsekas and Tsitsiklis, 1996; Sutton and Barto, 1998) for more information. The general idea is that a

good control for a control problem in discrete time is reached if for each of the finitely many time steps, one can learn efficiently the "expected cost-to-go" (also named Bellman-function or value-function in various fields of mathematical programming and computer science), i.e. a function that maps a state x and time step t to the expected cost (for an optimal strategy) starting from this state x at time t. This cost-to-go has to be learnt from examples. These examples can be sampled, and a procedure termed *Bellman-operator* (BO) provides the cost-to-go for each given example.

Roughly, the classical SDP algorithm, based on a sampler S and a learner L, is as follows. For each time step t, backwards from the last time step to the first: 1. S samples a finite set of examples $x_1,\ldots,x_n$ at time step t; 2. The BO (detailed below) compute the expected cost-to-go for each of these examples (using the previously learnt expected cost-to-go, i.e. the cost-to-go at time step $t+1$); 3. L learns the expected cost-to-go V_t at time step t from these examples.

The BO-procedure computes the cost-to-go at state x for time step t using simulations of the transition from time step t to time step $t+1$ and using the cost-to-go-function V_{t+1} at step $t+1$ as follows: (i) simulate each possible random outcome (ii) for each random outcome get by simulation the instantaneous cost and the future state y and use the cost-to-go $V_{t+1}(y)$ from y at time step $t+1$ (available thanks to the backward induction) for estimating the future-cost (iii) add these two costs for each random outcome, (iv) average all these results to get the required value, i.e. the expected cost-to-go $V_t(x)$.

The sampler is important in order to reduce the computational cost, a main issue of SDP. We here use tools that are not specific of SDP. More specific approaches for dynamic problems also exist; they are more orthogonal to, than comparable to our approaches below. (Barto et al., 1993) showed the importance of "reducing" the domain, when possible, by using simulations. If only a small subset of the full state space is interesting, this approach restricts the focus of the algorithm to the neighborhood of the visited states. A drawback is that you need an approximate solution before applying simulations, and you need simulations in order to reduce the domain and apply dynamic programming; therefore, this approach leads to complex unstable fixed-point iterations. However, this approach can deal with much bigger domains than the approach sampling the whole domain. (Thrun, 1992) studied how to avoid visiting many times the same area (leading to better guaranteed rates in some theoretical framework), and then reduce the curse of dimensionality.

3.2 Blind Active-Samplers

We present below point sets in $D=[0,1]^d$. P will be the notation for a set of points $P_1,\ldots,P_n$. $\#E$ is the cardinal of a set E. See e.g. (Tuffin, 1996; Owen, 2003; L'Ecuyer and Lemieux, 2002) for a general introduction to "well chosen" sets of points.

Low discrepancy. The most usual discrepancy is defined as follows:

$$Disc(P)=\sup_{r\in[0,1]}\left|area([0,r])-\frac{1}{n}\#\{i\in[[1,n]];P_i\leq r\}\right|$$

where $[0,r]$ is the set of q such that $\forall i\in[[1,d]]0\leq q_i\leq r_i$ and $area()$ is Lebesgue's-measure. Independent uniform random points have discrepancy roughly decreasing as $O(1/\sqrt{\#P})$. Well chosen deterministic or (non-naive) randomized points achieve $O(\log(\#P)^d/\#P)$. See `http://www.lri.fr/~teytaud/ldsfordplong.pdf` for further elements. We will in the sequel call "low-discrepancy sequence" a classical Niedereiter sequence (see e.g. (Niederreiter, 1992)). We will refer to this sequence as a QR (quasi-random) sequence in the sequel.

Low dispersion. Low dispersion is less widely used than low-discrepancy, but has some advantages (see e.g. discussions in (Lindemann and LaValle, 2003; LaValle and Branicky, 2002)). The most usual criterion (to be minimized) is

$$Dispersion(P)=\sup_{x\in D}\inf_{p\in P}d(x,p) \qquad (1)$$

where d is the euclidean distance. It is related to the following (easier to optimize numerically, except for some values of $\#P$) criterion (to be maximized) :

$$Dispersion_2(P)=\inf_{(x_1,x_2)\in D^2}d(x_1,x_2) \qquad (2)$$

A main difference is that optimizing eq. 2 "pushes" points on the frontier. This effect can be avoided as follows :

$$Dispersion_3(P)=\inf_{(x_1,x_2)\in D^2}d(x_1,\{x_2\}\cup D') \qquad (3)$$

where $D'=\{x\in\mathbb{R}^d;x\notin D\}$. We call "low-dispersion point set" (LD) a point set optimizing equation 2. We call "greedy-low-dispersion point set" (GLD) a point set optimizing equation 2 in a greedy manner, i.e. $P_1=(0.5,...,0.5)$ is the middle point of $[0,1]^d$, P_2 is such that $Dispersion_2(\{P_1,P_2\})$ is maximal, and P_n is such that $Dispersion_2(\{P_1,\ldots,P_{n-1},P_n\})$ is maximal. Our implementation is based on Kd-trees but Bkd-trees are possible (bkdtrees are similar to well-known kdtrees, but also allow fast online adding of

points; see (Procopiuc et al., 2002)). We call "greedy-low-dispersion point set far-from-frontier" (GLDfff) the equivalent point set with $dispersion_3$ instead of $dispersion_2$. We also tested the same dispersion with other L^p-distances than the euclidean one, without success. GLD is very successful, as shown in the sequel; we show a sample in figure 1.

Figure 1: A GLD-sample in dimension 2. Note that the random choice of points among various possible optimal points avoids (probably) too unequilibrated sets. The first points are shown with darker circles. There is randomness in the choice of the next point among a given gray-level.

3.3 A Non-blind Active-sampler

Many tools for active-learning exist (see introduction) but can not be used in our case as they are classification-specific or known moderately efficient in the regression case. We here propose the use of an evolutionary algorithm. This approach is new in the case of active learning and in the case of SDP, but it is inspired by evolutionary sampling (Liang and Wong, 2001). Its advantages are (i) it is user-friendly (ii) it works better than random and also than derandomized blind point sets at least for a non-negligible set of benchmarks.

Evolutionary algorithms (EA,(Baeck, 1995; Eiben and Smith, 2003)) are a classical tool for robustly optimizing a non-linear function without requiring derivatives. We here use them as a sampling algorithm, as follows: 1. generate an initial population uniformly on the domain; 2. evolve the population until the allowed number of fitness evaluations; 3. use as active sample the union of all offsprings of all epochs. Note that the learning algorithm is not embedded in this approach which is therefore quite general.

We define precisely below our EA, but we point out that any EA could be used as well. The parameters have not been specialized to our problem, we have used a tool that has been designed for optimization purposes and not for our particular application to sampling. Evolutionary algorithms exist for various forms of domains, continuous, discrete or mixed, and therefore our approach is quite general. Estimation of distribution algorithms could be used as well(Larranaga and Lozano, 2001). The EA that we use can be found in the *sgLibrary* (`http://opendp.sourceforge.net`). It implements a very simple genetic algorithm where the mutation is an isotropic Gaussian of standard deviation $\frac{\sigma}{\sqrt[d]{n}}$ with n the number of individuals and d the dimension of space. The crossover between two individuals x and y gives birth to two individuals $\frac{1}{3}x+\frac{2}{3}y$ and $\frac{2}{3}x+\frac{1}{3}y$. Let $\lambda_1,\lambda_2,\lambda_3,\lambda_4$ be such that $\lambda_1+\lambda_2+\lambda_3+\lambda_4=1$; at each generation :

1. we copy the $n\lambda_1$ best individuals (set S_1).
2. we combine the $n\lambda_2$ following best individuals with the individuals of S_1 (rotating among S_1 if $\lambda_1<\lambda_2$).
3. we mutate $n\lambda_3$ individuals among S_1 (again rotating among S_1 if $\lambda_1<\lambda_3$).
4. we randomly generate $n\times\lambda_4$ other individuals, uniformly on the domain.

The parameters are $\sigma=0.08,\lambda_1=1/10,\lambda_2=2/10,\lambda_3=3/10,\lambda_4=4/10$; these parameters are standard ones from the library and have not been modified for the work presented in this paper. The population size is $n=N^\alpha$, where $\alpha\in]0,1]$ is a parameter of the approach. Note that this algorithm is oriented towards sampling small values of the target function. Therefore, it is based on the idea that small values are more interesting. In the case of SDP, the target function is the sum of the instantaneous cost and the expected cost-to-go. Thus, this active sampling is efficient if we are dealing with a problem in which it is likely that trajectories are close to small values of this target function. We will see in our experiments that this is true in some problems, but not e.g. in stock management, leading to poor results in that case. We here see that the non-blind approach developed in this section looks appealing, but has robustness drawbacks. The tuning of α is a possible solution: $\alpha=1$ leads to the pure random sampling; smaller α leads to a more optimistic approach in which the sample is reinforced in parts for which costs are better (smaller), at the price of a weaker ability to avoid very large costs.

3.4 Experiments

The domain to be sampled is made of the continuous state space and a discrete state space. The samplers have then to sample a product $S\times M$, with S the continuous state space and $M=\{m_1,\ldots,m_k\}$ discrete and finite, being the exogenous Markov-Process. The following SM are used:

1. The SMs GLD, QR, LD, GLDfff are the blind approaches defined in section 3.2. The discrete parts of the state-space are sampled in a simple proportional manner: for sampling n points in $S\times M$, where S is the continuous state space and M is a discrete domain $M=\{m_1,\ldots,m_k\}$, a GLD, QR, LD or GLDfff point set is used for sampling S with $N=n/k$ points $x_1,\ldots,x_N$, and the sample is

$((x_1,m_1),\ldots,(x_N,m_1),\ldots,(x_1,m_k),\ldots,(x_N,m_k))$.

2. We have various random samplings. The SM RandomForAll denotes the pure blind (uniform, independent) random sampling of $S \times M$. The SM RandomOnlyForContinuousStates (ROFCS) denotes the pure blind random sampling of S, with a proportional sampling of M (i.e. m_1 appears the same number of times as m_2, ..., as m_k). The SM DeterministicForDim1 is equal to ROFCS, except that for each value m_i, the first coordinate of S is sampled by a regular grid (other coordinates are random). This is therefore deterministic if S is one-dimensional.

3. EAS-α denotes the non-blind approach using EA-sampling, as explained in section 3.3. The parameter is the α-parameter in section 3.3. All the experiments have been performed with the OpenDP library (Gelly and Teytaud, 2005) with the command line options "./runme.sh –nojava –nogui –test OptimizationExperiments –outputFile myResultsFile.txt –testedOptimizers '[[GeneticAlgorithmOptimizer]]' –testedRegressions '[[AutomaticRegression]]' –numExampleWanted X –nbPointsSamplingMethod 500 –nbRuns 44 –nbTryEvaluationsOptimization 70" where X$\in \mathbb{N}$ is the index of the problem. The parameters of the experiments are as follows: 500 points are sampled on the domain (same number of sampled points for each algorithm); all results are on 44 runs; the learner is the "AutomaticRegression" class from (Gelly and Teytaud, 2005), that uses the Gaussian-SVM of SVMTorch (Collobert and Bengio, 2001) and a heuristic rule choice of hyper-parameters. All problems are described in (Gelly et al., 2006) and in (Gelly and Teytaud, 2005) (including free downloads on `http://opendp.sourceforge.net`). The dimensionality of problems is the dimension of the continuous state space, excluding the discrete parts. The objective functions are to be minimized (lower values = better results).

Results about derandomized blind sampling. Results are summarized in table 1. Due to length constraints, all the details with confidence intervals, and results for other methods, are provided in `http://www.lri.fr/~teytaud/ldsfordplong.pdf`. The dimension refers to the dimension of the state space. Columns QR (resp. GLD, GLDfff), refer to the fact that QR-sampling (resp. GLD-sampling, GLDfff-sampling) outperforms the baseline random algorithm, namely ROFCS (in which the continuous part is pure independent random and the discrete part is proportional sampling); we also compared the algorithms to pure random sampling, which is usually worse than ROFCS (which can be seen as a very simple preliminar derandomization). For the column mentionning the fact that GLD is first-ranked, "**y**" stands for "GLD is significantly better than all other algorithms" and "y(=ST)" stands for "GLD is first-ranked and significantly better than all other algorithms except algorithm ST for which there's no statistically significant difference".

Results about non-blind active regression. We note EAS the algorithm with parameter α for non-blind active sampling of the state space defined in 3.3. In moderate dimension, the efficiency of EAS is very moderate; we here present results with high dimensionality and 500 points sampled per time step. We compared EAS, GLD and the best blind sampler on this problem. All results are averaged on 44 runs.

Due to length constraints, details for all problems are provided in `http://www.lri.fr/~teytaud/ldsfordplong.pdf` and we here show only results for the "Arm" and "Multi-Agent" problem; in summary for other problems: for Avoid (dim=8) and Walls (dim=8), GLD outperforms the EAS for any value of α; for Arm (dim=12) and Multi-agent (dim=8), EAS outperforms GLD for all values of α; for Shoot (dim=12) there's not significant difference in results; for Away (dim=8) the EAS outperform GLD moderately significantly or significantly

Table 1: Blind sampling methods. First, we see that GLD is often the best blind SM. It outperforms the random-sampling ROFCS in 7 out of 9 experiments. Second, other derandomized samplings are only sometimes better than random-sampling with proportional sampling for discrete parts of the state space; essentially, the difference between these other less-randomized samplings and ROFCS is not significant. The pure naive random sampling, RandomForAll (which includes random sampling of the discrete part also), is not presented in this table as it is meaningless in some cases (when there's no discrete part); its results, when meaningful, are poor. As a conclusion, (i) derandomizing the sampling for the discrete part is strongly better (of course, at least in problems for which there is a discrete part) (ii) GLD is a stable and efficient partially-derandomized sampler of continuous domains.

Problem (dim)	GLD 1st ranked	QR	GLD	GLDfff
WallsX4 (8)	**y**	y	y	n
AvoidX4 (8)	**y**	y	y	y
Stock (4)	**y**	n	y	n
Avoid (2)	y (=GLDfff)	y	y	y
Arm (3)	n	n	n	n
Walls (2)	**y**	y	y	n
Multiag. (8)	n	n	n	y
Shoot (3)	**y**	y	y	y
Away (2)	y (=QR)	n	y	n
Total	7/9	5/9	7/9	4/9

Table 2: Results for the "arm" and "multi-agent" problems. For the "multi-agent" problem, EAS is very efficient; this is probably related to the fact that intuitively, the criterion focusing on "good regions" (where the expected cost-to-go is small) is satisfactory for this problem: large bad regions of the domain are very unlikely thanks to the existence of reasonnable good paths. These large bad regions are explored by blind samplers. The criterion that we plug in EAS is relevant for this case, and should be adapted for other problems in order to get similar good results. EAS has a performance equivalent to ROFCS for the "arm" problem.

Arm dim.x4 = 12		Biagent dim.x4 = 8	
Sampler	Average score	Sampler	Average score
EAS-0.2	10.34±0.21	EAS-0.2	2295.23±16.08
EAS-0.3	10.17±0.20	EAS-0.3	2311.59±13.46
EAS-0.4	10.46±0.23	EAS-0.4	2179.09±15.62
EAS-0.5	10.62±0.21	EAS-0.5	2175.45±16.11
EAS-0.6	10.49±0.20	EAS-0.6	2156.59±15.66
EAS-0.7	10.54±0.23	EAS-0.7	2137.5±13.41
EAS-0.75	10.53±0.22	EAS-0.75	2167.95±14.26
EAS-0.8	10.37±0.20	EAS-0.8	2195±16.62
EAS-0.85	10.17±0.20	EAS-0.85	2184.09±13.91
EAS-0.9	10.44±0.20	EAS-0.9	2170.23±14.22
EAS-0.95	10.38±0.21	EAS-0.95	2245.91±12.39
ROFCS	10.39± 0.20	best-blind	
GLD	12.52 ±0.25	= GLD	2326.59 ±16.17

depending on α except for $\alpha = 0.95$.

4 CONCLUSION

This paper studies theoretically and practically active learning, in the case of regression. As pointed out in the introduction, the classification case is very different and non-blind approaches look much more appealing in that case. Our conclusions hold for regression and our experiments are performed in the SDP case in which robustness of sampling is particularly important.

It is sometimes argued that randomness provides robustness (Rust, 1997). We confirm in theorem 2.3 that derandomization should not be complete for the sake of robustness (strict determinism or almost determinism lead to the loss of universal consistency), but we show in theorem 2.4 that a strong derandomization, with only a moderate random part, is enough for UC. In particular, derandomized sequences as used in theorem 2.4 combine (i) known convergence rates (Cervellera and Muselli, 2003) of quasi-random sequences for learning smooth target functions and (ii) robustness of random sampling in terms of UC as in (Devroye et al., 1994).

This conclusion about the best "quantity" of randomness is exemplified in our experiments by, in some cases, the weakness of methods too strongly focusing on parts looking promising (EAS with $\alpha < 1$). On the other hand, these methods are sometimes very efficient for difficult cases (see Table 2). Also, we note that the best blind-method is GLD, often by far, and GLD is a very natural randomized low-dispersion sequence (see (Lindemann and LaValle, 2003; LaValle and Branicky, 2002) on this topic).

We believe in the importance of research about blind or non-blind active learning as active learning *is* a main bottleneck in reinforcement learning or SDP. We have the following positive conclusions for active regression:

1. Our **new blind sampling method**, namely GLD, outperforms all other blind samplers in most cases (see table 1). In particular, it is a natural extension to any number of points and any dimensionality of regular grid-samplings (see figure 1 for small number of points, different (and better than, at least in our experiments) from the low-dispersion approach LD, GLDfff, RandomForAll, QR, etc. This blind approach is the most stable one in our experiments (first ranked in most experiments); it has weaknesses in cases in which the frontier is not relevant. It is known that quasi-random-sequence have to be randomized for various no-bias properties (see e.g. (L'Ecuyer and Lemieux, 2002) about randomized quasi-Monte-Carlo sequences); in this paper we have (i) shown theoretical similar results for active learning pointing out that **randomness is necessary in active sampling** (ii) provided with GLD **a natural randomization** as we simply randomly pick up, for each point, one of the optimal points for the greedy criterion in equation 2 (see figure 1).

2. Our **new non-blind sampling method** is very easy to use; it is **easy to put expert knowledge in it** (increasing α for more carefully exploring the domain, reducing α for focusing on optimistic parts, changing the criterion, adding diversity in the evolutionary algorithm). However, experiments show that the criterion tested here is not universal; it works very efficiently for the "multi-agent" problem, providing strategies with smaller expected cost than all blind techniques tested, but not for some other problems. This is somewhat disappointing, but we guess that non-blind sampling methods do require some tuning. EAS moves continuously, thanks to the α-parameter, from random sampling (or possibly quasi-random sampling, but this has not been studied here) to a focus on areas verifying some used-defined criterion. Results appear robust w.r.t to α, but the criterion is probably a much stronger parameter; here, it focuses on areas with small costs; but for other problems it is probably much better to sample areas with large costs (e.g. when strong penalties can occur

when hitting boundaries). A main advantage of EAS is that it works in **high dimensionalities** in which very few published papers have good active-results.

3. In some cases, GLD is far better than GLDfff, but in one case GLD is the worst algorithm and GLDfff is (very significantly) the best one. As the main difference between these two algorithms is that GLD samples more strongly the frontier, this points out the simple fact that **the frontier of the state-space** can be very important: sometimes it is very relevant (e.g. stock management, in which marginal costs have a first approximation thanks to cost-to-go at corners) and sometimes it is pointless and expensive (GLD puts 2^d points on the corners among the $2^d + 1$ first points!).

REFERENCES

Baeck, T. (1995). *Evolutionary Algorithms in theory and practice*. New-York:Oxford University Press.

Barto, A., Bradtke, S., and Singh, S. (1993). Learning to act using real-time dynamic programming. Technical Report UM-CS-1993-002.

Bertsekas, D. and Tsitsiklis, J. (1996). Neuro-dynamic programming, athena scientific.

Cervellera, C. and Muselli, M. (2003). A deterministic learning approach based on discrepancy. In *Proceedings of WIRN'03, pp53-60*.

Chapel, L. and Deffuant, G. (2006). Svm viability controller active learning. In *Kernel machines for reinforcement learning workshop, Pittsburgh, PA*.

Cohn, D. A., Ghahramani, Z., and Jordan, M. I. (1995a). Active learning with statistical models. In Tesauro, G., Touretzky, D., and Leen, T., editors, *Advances in Neural Information Processing Systems*, volume 7, pages 705–712. The MIT Press.

Cohn, D. A., Ghahramani, Z., and Jordan, M. I. (1995b). Active learning with statistical models. In Tesauro, G., Touretzky, D., and Leen, T., editors, *Advances in Neural Information Processing Systems*, volume 7, pages 705–712. The MIT Press.

Collobert, R. and Bengio, S. (2001). Svmtorch: Support vector machines for large-scale regression problems. *Journal of Machine Learning Research*, 1:143–160.

Devroye, L., Gyorfi, L., Krzyzak, A., and Lugosi, G. (1994). the strong universal consistency of nearest neighbor regression function estimates.

Eiben, A. and Smith, J. (2003). *Introduction to Evolutionary Computing*. springer.

Gelly, S., Mary, J., and Teytaud, O. (2006). Learning for dynamic programming, proceedings of esann'2006, 7 pages, http://www.lri.fr/∼teytaud/lfordp.pdf.

Gelly, S. and Teytaud, O. (2005). Opendp, a c++ framework for stochastic dynamic programming and reinforcement learning.

Kearns, M., Mansour, Y., and Ng, A. (1999). A sparse sampling algorithm for near-optimal planning in large markov decision processes. In *IJCAI*, pages 1324–1231.

Larranaga, P. and Lozano, J. A. (2001). *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*. Kluwer Academic Publishers.

LaValle, S. and Branicky, M. (2002). On the relationship between classical grid search and probabilistic roadmaps. In *Proc. Workshop on the Algorithmic Foundations of Robotics*.

L'Ecuyer, P. and Lemieux, C. (2002). Recent advances in randomized quasi-monte carlo methods.

Lewis, D. and Gale, W. (1994). Training text classifiers by uncertainty sampling. In *Proceedings of International ACM Conference on Research and Development in Information Retrieval*, pages 3–12.

Liang, F. and Wong, W. (2001). Real-parameter evolutionary sampling with applications in bayesian mixture models. *J. Amer. Statist. Assoc.*, 96:653–666.

Lindemann, S. R. and LaValle, S. M. (2003). Incremental low-discrepancy lattice methods for motion planning. In *Proceedings IEEE International Conference on Robotics and Automation*, pages 2920–2927.

Munos, R. and Moore, A. W. (1999). Variable resolution discretization for high-accuracy solutions of optimal control problems. In *IJCAI*, pages 1348–1355.

Niederreiter, H. (1992). *Random Number Generation and Quasi-Monte Carlo Methods*.

Owen, A. (2003). *Quasi-Monte Carlo Sampling, A Chapter on QMC for a SIGGRAPH 2003 course*.

Procopiuc, O., Agarwal, P., Arge, L., and Vitter, J. (2002). Bkd-tree: A dynamic scalable kd-tree.

Rust, J. (1997). Using randomization to break the curse of dimensionality. *Econometrica*, 65(3):487–516.

Schohn, G. and Cohn, D. (2000). Less is more: Active learning with support vector machines. In Langley, P., editor, *Proceedings of the 17th International Conference on Machine Learning*, pages 839–846. Morgan Kaufmann.

Seung, H. S., Opper, M., and Sompolinsky, H. (1992). Query by committee. In *Computational Learning Theory*, pages 287–294.

Sloan, I. and Woźniakowski, H. (1998). When are quasi-Monte Carlo algorithms efficient for high dimensional integrals? *Journal of Complexity*, 14(1):1–33.

Sutton, R. and Barto, A. (1998). *Reinforcement learning: An introduction.* MIT Press., Cambridge, MA.

Thrun, S. B. (1992). Efficient exploration in reinforcement learning. Technical Report CMU-CS-92-102, Pittsburgh, Pennsylvania.

Tuffin, B. (1996). On the use of low discrepancy sequences in monte carlo methods. In *Technical Report 1060, I.R.I.S.A.*

Vidyasagar, M. (1997). *A Theory of Learning and Generalization, with Applications to Neural Networks and Control Systems*. Springer-Verlag.

GENETIC REINFORCEMENT LEARNING OF FUZZY INFERENCE SYSTEM APPLICATION TO MOBILE ROBOTIC

Abdelkrim Nemra, Hacene Rezine and Abdelkrim Souici
Unit of Control, Robotic and Productic Laboratory
Polytechnical Military School
{karim_nemra, Rezine_hacene_emp, aks752005}@yahoo.fr

Keywords: Reinforcement learning, Fuzzy controllers' Genetic algorithm, mobile robotic.

Abstract: An efficient genetic reinforcement learning algorithm for designing Fuzzy Inference System (FIS) with out any priory knowledge is proposed in this paper. Reinforcement learning using Fuzzy Q-Learning (FQL) is applied to select the consequent action values of a fuzzy inference system, in this method, the consequent value is selected from a predefined value set which is kept unchanged during learning and if the optimal solution is not present in the randomly generated set, then the performance may be poor. Also genetic algorithms (Genetic Algorithm) are performed to on line search for better consequent and premises parameters based on the learned Q-values as adaptation function. In Fuzzy-Q-Learning Genetic Algorithm (FQLGA), memberships (premises) parameters are distributed equidistant and the consequent parts of fuzzy rules are randomly generated. The algorithm is validated in simulation and experimentation on mobile robot reactive navigation behaviors.

1 INTRODUCTION

In the last decade, fuzzy logic has supplanted conventional technologies in some scientific applications and engineering systems especially in control systems, particularly the control of the mobile robots evolving (moving) in completely unknown environments. Fuzzy logic has the ability to express the ambiguity of human thinking and translate expert knowledge into computable numerical data. Also, for real-time applications, its relatively low computational complexity makes it a good candidate. A fuzzy system consists of a set of fuzzy if-then rules. Conventionally, the selection of fuzzy if-then rules often relies on a substantial amount of heuristic observation to express the knowledge of proper strategies

Recently, many authors proved that it is possible to reproduce the operation of any standard continuous controller using fuzzy controller (Jouffe, 1996), (Watkins, 1992), (Glorennec, 1997), (Dongbing, 2003). However it is difficult for human experts to examine complex systems, then it isn't easy to design an optimized fuzzy controller. Generally the performances of a system of fuzzy inference (SIF) depend on the formulation of the rules, but also the numerical specification of all the linguistic terms used and an important number of choices is *given a priori* also it is not always easy or possible to extract these data using human expert. These choices are carried with empirical methods, and then the design of the FIS can prove to be long and delicate vis-à-vis the important number of parameters to determine, and can lead then to a solution with poor performance. To cope with this difficulty, many researchers have been working to find learning algorithms for fuzzy system design. These automatic methods enable to extract information when the experts' *priori* knowledge is not available.

The most popular approach to design FLC may be a kind of supervised learning where the training data is available. However in real applications extraction of training data is not always easy and become impossible when the cost to obtain training data is expensive. For these problems, reinforcement learning is more suitable than supervised learning. In reinforcement learning, an agent receives from its environment a critic, called reinforcement, which can be thought of as a reward or a punishment. The objective then is to generate a policy maximizing on average the sum of the rewards in the course of time, starting from experiments (state, action, reward).

This paradigm corresponds to one of the fundamental objectives of the mobile robotics which constitutes a privileged applicability of reinforcement learning. The paradigm suggested is to regard behaviour as a sensor-effectors correspondence function. The objective is to favour the robots autonomy using learning algorithms.

In this article, we used the algorithm of reinforcement learning, Fuzzy Q-Learning (FQL) (Jouffe, 1996), (Souici, 2005) which allows the adaptation of apprentices of the type SIF (continuous states and actions), fuzzy Q-learning is applied to select the consequent action values of a fuzzy inference system. For these methods, the consequent value is selected from a predefined value set which is kept unchanged during learning, and if an improper value set is assigned, and then the algorithm may fail. Also, the approach suggested called Fuzzy-Q-Learning Genetic Algorithm (FQLGA), is a hybrid method of Reinforcement Genetic combining FQL and genetic algorithms for on line optimization of the parametric characteristics of a SIF. In FQLGA we will tune free parameters (precondition and consequent part) by genetic algorithms (GAs) which is able to explore the space of solutions effectively.

This paper is organized as follows. In Section 2, overviews of Reinforcement learning, implementation and the limits of the Fuzzy-Q-Learning algorithm is described. The implementation and the limits of the Fuzzy-Q-Learning algorithm are introduced in Section 3. Section 4 describes the combination of Reinforcement Learning (RL) and genetic algorithm (GA) and the architecture of the proposed algorithm called Fuzzy-Q-Learning Genetic Algorithm (FQLGA). This new algorithm is applied in the section 5 for the on line learning of two elementary behaviors of mobile robot reactive navigation, "Go to Goal" and "Obstacles Avoidance". Finally, conclusions and prospects are drawn in Section 6.

2 REINFORCEMENT LEARNING

As previously mentioned, there are two ways to learn either you are told what to do in different situations or you get credit or blame for doing good respectively bad things. The former is called supervised learning and the latter is called learning with a critic, of which reinforcement learning (RL) is the most prominent representative. The basic idea of RL is that agents learn behaviour through trial-and-error, and receive rewards for behaving in such a way that a goal is fulfilled.

Reinforcement signal**,** measures the utility of the exits suggested relative with the task to be achieved, the received reinforcement is the sanction (positive, negative or neutral) of behaviour: this signal states that it should be done without saying how to do it. The goal of reinforcement learning is to find the behavior most effective, i.e. to know, in each possible situation, which action is achieved to maximize the cumulated future rewards.Unfortunately the sum of rewards could be infinite for any policy. To solve this problem a discount factor is introduced.

$$R = \sum_{k \neq 0}^{\infty} \gamma^{k} r_{k} \qquad (1)$$

Where $0 \leq \gamma \leq 1$ is the *discount factor.*

The idea of RL can be generalized into a model, in which there are two components: an agent that makes decisions and an environment in which the agent acts. For every time step, the agent perceives information from the environment about the current state, ***s***. The information perceived could be, for example, the positions of a physical agent, to simplify say the ***x*** and ***y*** coordinates. In every state, the agent takes an action $\boldsymbol{u_t}$, which transits the agent to a new state. As mentioned before, when taking that action the agent receives a reward.

Formally the model can be written as follows; for every time step ***t*** the agent is in a state $\boldsymbol{st} \in S$ where *S* is the set of all possible states, and in that state the agent can take an action $at \in (At)$, where (At) is the set of all possible actions in the state ***st***. As the agent transits to a new state $st+1$ at time $\boldsymbol{t+1}$ it receives a numerical reward $\boldsymbol{rt+1}$. It up to date then its estimate of the function of evaluation of the action using the immediate reinforcement, $\boldsymbol{rt_{+1}}$, and the estimated value of the following state, $Vt\ (\boldsymbol{St_{+1}})$, which is defined by:

$$V_t\left(s_{t+1}\right) = \max_{u_t \in U_{t+1}} Q\left(s_{t+1}, u_t\right) \qquad (2)$$

The Q-value of each state/action pair is updated by:

$$Q\left(s_{t+1}, u_t\right) = Q\left(s_t, u_t\right) + \beta\left\{r_{t+1} + \gamma V_t\left(s_{t+1}\right) - Q\left(s_t, u_t\right)\right\} \quad (3)$$

Where $r_{t+1} + \gamma V_t\left(s_{t+1}\right) - Q\left(s_t, u_t\right)$ the TD error and β is the learning rate.

This algorithm is called Q-Learning. It shows several interesting characteristics. The estimates of the function Q, also called the Q-values, are independent of the policy pursued by the agent. To calculate the function of evaluation of a state, it is not necessary to test all the possible actions in this state but only to take the maximum Q-value in the new state (eq.4). However, the too fast choice of the action having the greatest Q-value:

$$u_t = \arg\max_{u_t \in U_{t+1}} Q(s_{t+1}, u_t) \quad (4)$$

can lead to local minima. To obtain a useful estimate of Q, it is necessary to sweep and evaluate the whole of the possible actions for all the states: it is what one calls the phase of exploration (Jouffe, 1996), (Souici, 2005). In the preceding algorithm, called TD (0), we use only the state which follows the robot evolution, Moreover only the running state is concerned. Sutton (Souici, 2005) extended the evaluation in all the states, according to their eligibility traces that memorise the previously visited state action pairs in our case. Eligibility traces can be defined in several ways (Jouffe, 1996), (Watkins, 1992), (Souici, 2005). Accumulating eligibility is defined by:

$$e_t(s) = \begin{cases} 1 + \gamma\lambda e_{t-1}(s) & si \quad s = s_t \\ \gamma\lambda e_{t-1}(s) & \text{sinon} \end{cases} \quad (5)$$

The algorithm Q (λ) is a generalization of Q-Learning which uses the eligibilities truces (Souici, 2005): *Q-Learning* is then a particular case of Q (λ) when λ=0.

$$Q(s_{t+1}, u_t) = Q(s_t, u_t) + \beta\left\{r_{t+1} + \gamma V_t(s_{t+1}) - Q(s_t, u_t)\right\} e_t(s) \quad (6)$$

3 FUZZY Q-LEARNING ALGORITHM

In mobile robotics, input and output variables given by the sensors and effectors are seldom discrete, or, if they are, the number of state is very large. However reinforcement learning such as we described it uses a discrete space of states and actions which must be have reasonable size to enable the algorithms to converge in an acceptable time in practice.

The principle of the Fuzzy Q-Learning algorithm consists in choosing for each rule R_i a conclusion among a whole of actions available for this rule. Then it implements a process of competition between actions. With this intention, the actions corresponding to each rule R_i have a quality q^i which then determines the probability to choose the action. The output action (*discrete* or *continues*) is then the result of the inference between various actions locally elected. The matrix q enables to implement not only the local policies the rules, but also to represent the function of evaluation of the overall t-optimal policy.

For every time step t the agent is in a state $st \in S$ where S is the set of all possible states, and in that state the agent can take an action $at \in (At)$, where (At) is the set of all possible actions in the state st. As the agent receives a numerical reward $rt+1 \in R$ at time $t + 1$, and it transits to a new state $st+1$. It then perceives this state by the means of its activated degree of its rules. The algorithm FQL uses a Policy of Exploration/Exploitation (PEE) (Jouffe, 1996), (Souici, 2005), combining a random exploration part $\rho^i(U)$ and determinist guided part $\eta^i(U)$.

The steps are summarized as follows:

1. Calculate of the evaluation function:

$$Q_t^*(S_{t+1}) = \sum_{R_i \in A_t} (Max_{U \in U}(q_t^i(U)) \alpha_{R_i}(S_{t+1}),$$

2. Calculate the TD error

$$\tilde{\varepsilon}_{t+1} = r_{t+1} + \gamma Q_t^*(S_{t+1}) - Q_t(S_t, U_t)$$

3. Update the matrix of Q values

$$q_{t+1}^i = q_t^i + \beta.\tilde{\varepsilon}_{t+1} e_t^{i^T}, \forall R_i$$

4. Election of the action to be applied

$$U_{t+1}(S_{t+1}) = \sum_{R_i \in A_{t+1}} Election_U(q_{t+1}^i) \alpha_{R_i}(S_{t+1}), \forall U \in U,$$

Where Election is defined by:

$$Election_U(q_{t+1}^i) = ArgMax_{U \in U}(q_{t+1}^i(U) + \eta^i(U) + \rho^i(U))$$

5. Update of the eligibility traces

$$e_{t+1}^i(U^i) = \begin{cases} \gamma\lambda e_t^i(U^i) + \phi_{t+1}^i, (U^i = U_{t+1}^i) \\ \gamma\lambda e_t^i(U^i), \quad \text{sinon} \end{cases}$$

And new calculation of the evaluation function.

$$Q_{t+1}(S_{t+1}, U_{t+1}) = \sum_{R_i \in A_{t+1}} q_{t+1}^i(U_{t+1}^i) \alpha_{R_i}(S_{t+1}),$$

This value will be used to calculate the TD error in the next step time. However the performances of the controller are closely dependent on the correct choice of the discrete actions set, witch is determined using *a priori* knowledge about system, for complex systems like robots, *priori* knowledge are not available, then it becomes difficult to determine a set of correct actions in which figure the optimal action for each fuzzy rule. To solve this problem and to improve the performances of the

reinforcement learning, the genetic algorithms will explore the broadest space of solutions to find the solution optimal (Dongbing, 2003), (Chia-Feng, 2005), (Min-Soeng, 2000), (Chia-Feng, 2003), and that without any priori knowledge.

4 GENETIC REINFORCEMENT ALGORITHM

Genetic Algorithms (GA) are stochastic optimization algorithms, founded on species evolution mechanisms (Goldberg, 1994). In GA, a candidate solution for a specific problem is called an individual or a chromosome and consists of a linear list of genes. Each individual represents a point in the search space and, hence, a possible solution to the problem. A population consists of a finite number of individuals. Each individual is decided by an evaluating mechanism to obtain its fitness value. Based on this fitness value and undergoing genetic operators, a new population is generated iteratively, with each successive population referred to as a generation.

Genetic Reinforcement Learning enable to determine the best set of parameters of (antecedents / consequences) starting from a random initialization of these parameters and in each iteration, only one action is applied on the system to the difference of a traditional genetic algorithm GA use three basic operators (the selection, crossover, and mutation) to manipulate the genetic composition of a population:

- *Reproduction*: Individuals are copied according to their fitness values. The individuals with higher fitness values have more offspring than those with lower fitness values.
- *Crossover*: The crossover will happen for two parents that have high fitness values with the crossover probability p_c. One point crossover is used to exchange the genes.
- *Mutation*: The real value mutation is done by adding a certain amount of noise (Gaussian in this paper) to new individuals to produce the offspring with the mutation probability p_m. For the ***ith*** variable in ***jth*** individual, it can be expressed as:

$$a_{t+1} = a_t + \beta(i).N(0,\sigma) \qquad (7)$$

Where N denote a Gaussian noise, and $\beta(i) = \exp(-i)$ for the i^{th} generation.

4.1 FQLGA Algorithm

Because of its simplicity, a Takagi-Sugeno Fuzzy inference system is considered, with triangular membership function. The structure (partition of its input space, the determination of the number of *IF-THEN* rules) of the SIF is predetermined.

Rule R_i *:if* S_1 *is* L^i_1 *and.....and* S_{Ni} *is* L^i_{Ni} *then* Y *is* a^i_j *with* q^i_j

a^i_j is a vector representing the discrete set of K conclusions generated randomly for the rule R_i with which is associated a vector q^i_j representing the quality of each action (*i= 1 ~N and j= 1 ~ K).*

The principle of the approach is to use a population of K (SIF) and to treat the output of each one of them as a possible action to apply on the system. FQL algorithm exploits local quality function q witch is associated with each action of a fuzzy rule (équat.6) whereas FQLGA algorithm uses as function fitness the sum of local qualities q given by:

$$f(Ind_j) = Q(S_{t+1}, SIF_{t+1}) = \sum_{i=1}^{N} q^i_j \qquad (8)$$

To reduce the training time, the quality matrix Q is not initialized after each iteration but undergoes the same genetic operations as those applied to the set of the individuals (selection, crossing).

4.2 Optimization of the Consequent Part of a FIS

A population of K individuals of a predetermined structure is adopted. The size of an individual is equal to number N of the FIS's rules. The architecture of FQLGA algorithm proposed for the optimization of the conclusions is represented on the figure (1).

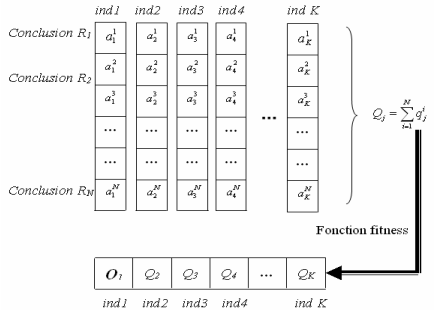


Figure 1:Representation of the individuals and qualities of the actions in FQLGA algorithm.

4.3 Optimization of the Antecedent Part of a SIF

To find the best set of premises generated by GA, a population made up of *M* SIF is created. Each individual (FIS) of the population encode the parameters of the antecedents i.e. the modal points of the FIS and his performance is evaluated by the fitness function of *Q* (global quality).

The conclusion part of each individual SIF remains fixed and corresponds to the values determined previously. The coding of the membership functions of the antecedent part of a FIS (individual) is done according to the figure (2). To keep the legibility of the SIF, we impose constraints during the evolution of the FIS to ensure the interpretability of the FIS.

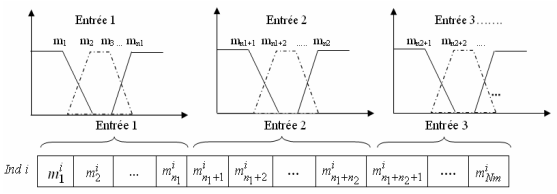


Figure 2: Coding of the parameters of the antecedent part.

Input N : $m^1_{N_m - n_N + 1} < .... < m^1_{Nm-2} < m^1_{Nm}$

The fitness function used in by genetic algorithm for the optimization of the antecedent part is the global quality of the FIS which uses the degree of activation of the fuzzy rules; this fitness function is given by the following equation:

$$f(ind_i) = Q(S(t), SIF_i) = \frac{\sum_{R_i \in R_A} \alpha_{Ri}(S(t)).q_i^i(t)}{\sum_{R_i \in R_A} \alpha_{R_i}(S(t))} \quad (9)$$

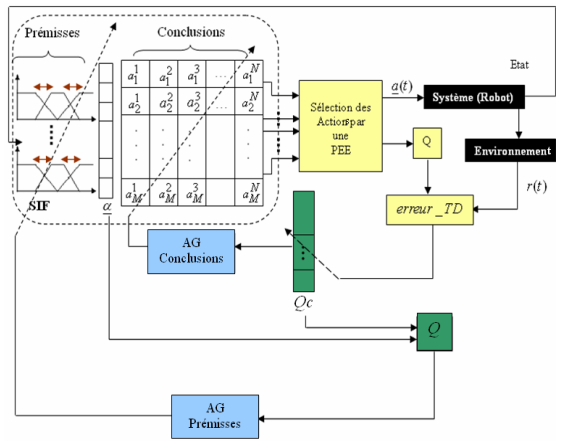


Figure 3: General architecture of FQLGA algorithm.

4.4 Optimization of the Consequent and Antecedent Part of FIS

A SIF consists of a set of fuzzy rules each one of it is made of an antecedent and a consequent part. Optimize the antecedent and the consequent part of the fuzzy controller at the same time is a complex problem which can admit several solutions which are not acceptable in reality, the great number of possible solutions makes the algorithm very heavy and can lead to risks of instability.

FQLGA algorithm proposed in (fig.3) for the optimization of the premises and the conclusions allows the total parametric optimization of the FIS in three stages represented in the flow chart (fig.4).

- At the beginning of the learning process, the quality matrix is initialized at zero, and then traditional algorithm FQL evaluates each action using an exploration policy. This step finishes when a number of negative reinforcements is received.
- After the evaluation of the individuals, the genetic algorithm for the optimization of the consequent part of the fuzzy rules creates a new better adapted generation. This stage is repeated until obtaining convergence of the conclusions or after having reached a certain number of generations. The algorithm passes then to the third stage:
- Once the conclusions of the FIS are optimized, the second genetic algorithm for the optimization of the antecedent part is carried out to adjust the positions of the input membership functions of the controller which are initially equidistant on their universe of discourse.

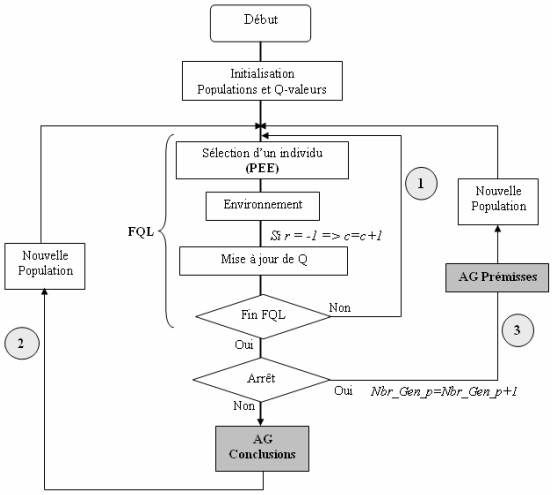


Figure 4: Flow chart of FQLGA algorithm.

5 EXPERIMENTAL RESULTS

To verify the performance of FQLGA two elementary behaviors of reactive navigation of a mobile robot: "Go to *Goal"* and "*Obstacles Avoidance"* are presented in this section; the algorithm was adopted in the experiments for both simulations (Saphira simulator) and real robots (Pioneer II).

5.1 «Go to Goal» Behaviour

The two input variables are: the angle"θ_{Rb}" between the robot velocity vector and the robot-goal vector, and the distance robot-goal "ρ_b".They are respectively defined by three (*Negative, Zeros, Positive*) and two (*Near, Loin*) fuzzy subsets (fig.5). The two output variables are the rotation speed, *Vrot_CB* and the translation speed *Vtran_CB* each output is represented by nine actions initialised randomly.

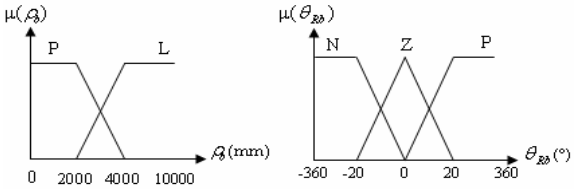


Figure 5: Membershipfunctions of the input space.

The reinforcement functions adopted for the two outputs are respectively given by:

$$r_{Vrot\_CB}(t) = \begin{cases} 1 & Si \quad (\theta \cdot \dot{\theta} < 0 \ ou \ -1° < \theta < +1°) \\ 0 & Si \quad (\theta \cdot \dot{\theta} = 0) \\ -1 & Else \end{cases}$$

$$r_{Vtran\_CB}(t) = \begin{cases} 1 & Si \quad Vtrans \le 0.15\rho_0 \text{ et } \rho_0 \ge 800 \\ 1 & Si \quad Vtrans \ge 220 \text{ et } \rho_0 \ge 800 \text{ et } abs(\theta) < 5 \\ -1 & Else \end{cases} \qquad (10)$$

The parameters of FQL algorithm and the genetic algorithms are as follows:

γ	λ	L_p	L_c	N_p	N_c	p_m
0.9	0.7	5	6	10	09	0.2

LP and *Lc* respectively indicate the sizes of the chromosomes for the antecedents and the conclusions part, *Np, Nc* respectively represent the size of the population of the parameters of antecedent and the conclusions and *Pm* the probability of mutation.

The simulation results are given of the first behaviour "Go to Goal" are presented in the figure (6), 28 generations were sufficient to find the good actions.

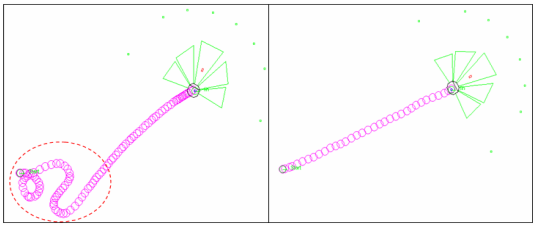


Figure 6: Go to goal: Training/Validation.

Figure (7) shows the convergence of the fitness values of the genetic algorithms for the two output variables *Vrot_CB* and *Vtran_CB* obtained during experimental test.

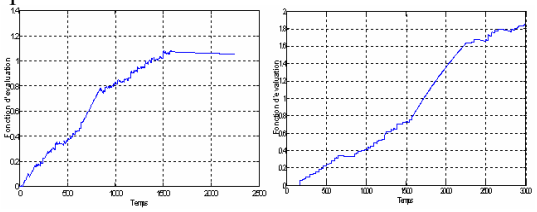


Figure 7: Fitness functions for the two output variables.

5.2 «Obstacles Avoidance» Behaviour

The Inputs of the FIS are the distances provided by the ultrasounds sensors to in the three directions (Frontal, Left and Right) and defined by the three fuzzy subsets: near (N), means (M), and far (F) (fig.8).

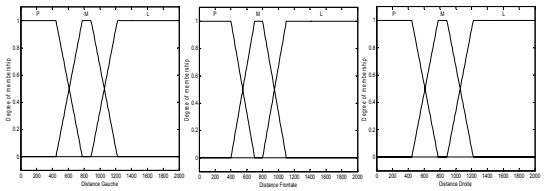


Figure 8:Memberships Functions of the inputs variables.

The conclusions (rotation speeds) are initialized randomly. The translation speed of *Vtran_EO* is given analytically; it is linearly proportional to the frontal distance:

$$Vtran\_EO = \frac{V\max}{D\max}.(Dis\_F - Ds) \qquad (11)$$

V max is the maximum speed of the robot equal to 350mm/s.

$D\max$ is the maximum value measured by the frontal sensors and estimated to 2000mm and Ds is a safe distance witch is fixed at 250mm.

The function of reinforcement is defined as follows (Souici, 2005):

$$r_{Vrot\_CB}(t) = \begin{cases} 1\times Signe\ (Vrot\_EO) \quad if\ \ Dis\_R < Dis\_L \\ \quad or\ \ Dis\_F < Dis\_L \quad and\ \ D\min<800 \\ -1\times Signe\ (Vrot\_EO)\ and\ \ Dis\_L < Dis\_R \\ \quad or\ \ Dis\_F < Dis\_R and\ \ D\min<800 \\ 0 \quad elsewhere \end{cases} \tag{12}$$

with $Dmin=min\ (Dis\_L, Dis\_F, Dis\_R)$

The parameters of FQL algorithm and the genetic algorithms are identical to the preceding behaviour except for the sizes of the chromosomes Lp=12 and Lc=27.

Figure (9) represents the trajectories of the robot during the learning phase with FQL algorithm and a random initialization of the consequents part for the 27 fuzzy rules. Several situations of failures are observed, this example show the limits of traditional FQL algorithm when priory knowledge about the system are not available.

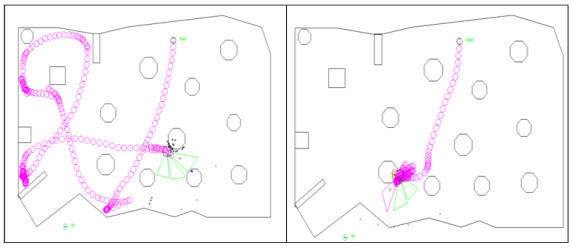


Figure 9: Trajectories of the robot obtained by FQL algorithm using a random initialization of parameters.

The trajectories of figure (10) show the effectiveness of the association of the reinforcement learning FQL and the genetic algorithm as stochastic tool for exploration. FQLGA Algorithm enables to find an optimal FIS for the desired behaviour (obstacles avoidance). The duration of learning depend to the genetic algorithm parameters and the obstruction density of the environment. We observe that after each generation the quality of the FIS (sum of local qualities) increases, which give more chance to the best individuals to be in the next generations.

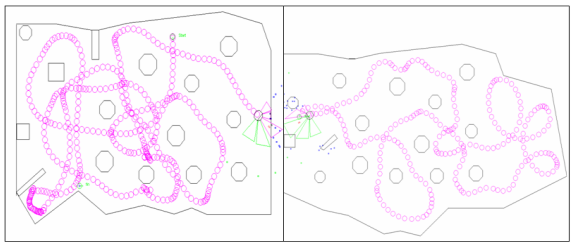


Figure 10: Learning/Validation Trajectories of the robot with FQLGA algorithm for various environments.

Figure (11) shows the performances of FQLGA algorithm compared to the FQL algorithm which can be blocked in a local minimum when the optimal solution is not present in the randomly generated set of actions. On the other hand FQLGA algorithm converges towards the optimal solution independently of the initialized values.

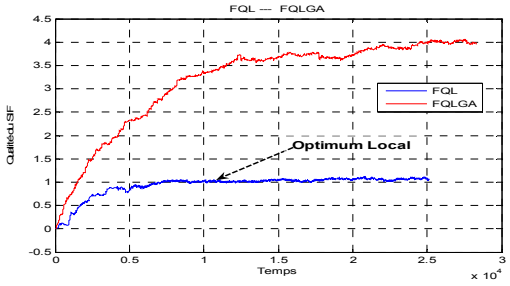


Figure 11: Evolution of the quality of the Fuzzy controller with FQL and FQLGA algorithms.

5.3 Experimental Results with the Real Robot Pioneer II

Figure (12) represents the results of the on line learning of the robot Pioneer II for the behaviour "Go to goal". During the learning phase, the robot does not follow a rectilinear trajectory (represented in green) between the starting point and the goal point because several actions are tested (exploration). Finally the algorithm could find the good actions, and the robot converges towards the goal marked in red colour, the necessary time to find these good actions is estimated at 2mn. Each generation is generated after having noted twenty (20) failures. The learning process requires respectively 32 and 38 generations for GA to determine rotation and translation speed.

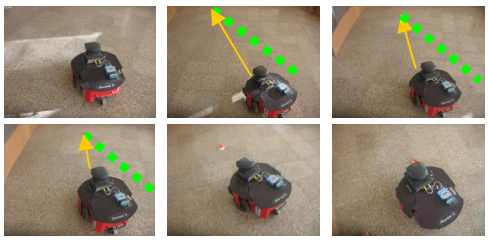


Figure 12: On line learning of the real robot Pioneer II, "Go to goal" behaviour.

Figure (13) represents the results of the on line learning of the "Obstacles Avoidance" robot behaviour. For reasons of safety, we consider that the minimal distances detected by the frontal sonars and of left/right are respectively 408 mm and of 345 mm a lower value than these distances is considered then as a failure and involves a retreat (represented in green) of the mobile robot. A generation is created after 50 failures. The genetic algorithms require 14 generations to optimize the conclusions and 24 generations to optimize the parameters of the antecedents. The duration of on line learning is estimated at 20 min, this time is acceptable vis-à-vis the heaviness of the traditional genetic algorithms.

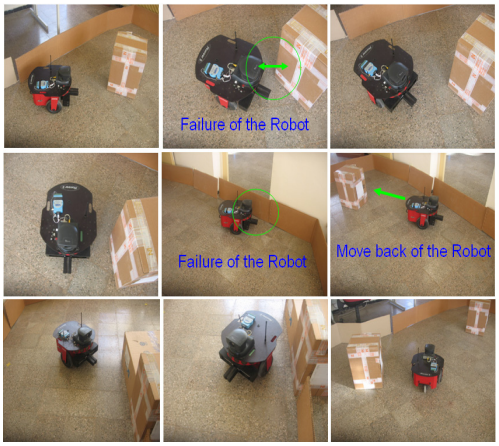


Figure 13: On line learning of the real robot Pioneer II, behaviour "Obstacle Avoidance».

Figure (14) represents the evolution of the fitness function obtained during this experimentation.

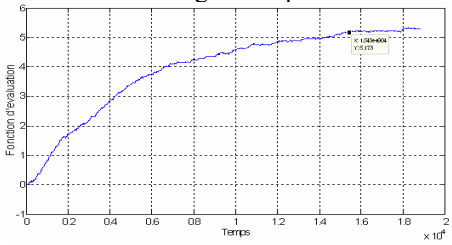


Figure 14: Fitness function evolution, "Obstacles avoidance" robot behaviour.

6 CONCLUSION

The combination of the reinforcement Q-Learning algorithm and genetics Algorithms give a new type of hybrid algorithms (FQLGA) which is more powerful than traditional learning algorithms. FQLGA proved its effectiveness when no *priori* knowledge about system is available. Indeed, starting from a random initialization of the conclusions values and equidistant distribution for the membership functions for antecedent part the genetic algorithm enables to find the best individual for the task indicated using only the received reinforcement signal. The controller optimized by FQLGA algorithm was validated on a real robot and satisfactory results were obtained. The next stage of this work is the on line optimization of the structure of the Fuzzy controller.

REFERENCES

L. Jouffe, "Actor-Critic Learning Based on Fuzzy Inference System", Proc of the IEEE International Conference on Systems, Man and Cybernetics. Beijing, China, pp.339-344, 1996.

C.Watkins, P. Dayan: "Q-Learning Machine Learning", pp.279-292, 1992.

P. Y. Glorennec and L. Jouffe, "Fuzzy Q-learning," Proc. Of IEEE Int. Con On Fuzzy Systems, pp. 659-662, 1997

Dongbing Gu, Huosheng Hu, Libor Spacek " Learning Fuzzy Logic Controller for Reactive Robot Behaviours" Proc. of IEEE/ASME International Conference on Advanced Japan, pp.20-24 July, 2003.

A. Souici **:** **"**Apprentissage par Renforcement des Systèmes d'inférence Floue par des Méthodes par renforcement application à la navigation réactive d'un robot mobile**",** Thèse de Magistère, Ecole Militaire Polytechnique, Janvier 2005.

Chia-Feng Juang " Combination of online Clustering and Q-Value Based GA for Reinforcement Fuzzy System Design" IEEE Transactions On Systems, Man, And Cybernetics Vol. 13, N°. 3, pp.289-302, June 2005

Min-Soeng Kim and Kim and Ju-Jang Lee. "*Constructing a Fuzzy Logic ControllerUsing Evolutionary Q-Learning"* IEEE. pp.1785-1790, 2000

Chia-Feng Juang and Chun-Feng Lu. Combination of On-line Clustering and Q-value Based Genetic Reinforcement Learning For Fuzzy Network Design 0-7803-7898-9/03/$17.00 0 2003 IEEE.

David Goldberg "A*lgorithme génétique Exploration, optimisation et apprentissage automatique"*» Edition Addison-Wesley, France 1994.

PIECEWISE CONSTANT REINFORCEMENT LEARNING FOR ROBOTIC APPLICATIONS

Andrea Bonarini, Alessandro Lazaric and Marcello Restelli
Department of Electronics and Information, Politecnico di Milano
piazza Leonardo Da Vinci, 32
20133, Milan, Italy
{*bonarini,lazaric,restelli*}*@elet.polimi.it*

Keywords: Robot Learning, Reinforcement Learning.

Abstract: Writing good behaviors for mobile robots is a hard task that requires a lot of hand tuning and often fails to consider all the possible configurations that a robot may face. By using reinforcement learning techniques a robot can improve its performance through a direct interaction with the surrounding environment and adapt its behavior in response to some non-stationary events, thus achieving a higher degree of autonomy with respect to pre-programmed robots. In this paper, we propose a novel reinforcement learning approach that addresses the main issues of learning in real-world robotic applications: experience is expensive, explorative actions are risky, control policy must be robust, state space is continuous. Preliminary results performed on a real robot suggest that on-line reinforcement learning, matching some specific solutions, can be effective also in real-world physical environments.

1 INTRODUCTION

A lot of research efforts have been spent in robotics to identify control architectures with the aim of making the writing of control programs easier. Although many advances have been made, it is often difficult for a programmer to specify how to achieve the desired solution. Furthermore, it is hard to take into consideration all the possible configurations the robot may face or the changes that may occur in the environment.

Reinforcement Learning (RL) (Sutton and Barto, 1998) is a well-studied set of techniques that allow an agent to achieve, by trial-and-error, optimal policies (i.e., policies that maximize the expected sum of observed rewards) without any a priori information about the problem to be solved. In the RL paradigm, the programmer, instead of programming how the robot should behave, has just to specify a reward function that models how good is an action when taken in a given state. This level of abstraction allows to write specifications for the robot behavior in a short time and to obtain better and more robust policies with respect to hand-written control code.

Despite the huge research efforts in the RL field, the application of RL algorithms to real-world robotic problems is quite limited. The difficulty to gather experience, the necessity to avoid dangerous configurations, the presence of continuous state variables are some of the features that make the application of RL techniques to robotic tasks complex.

In this paper, we analyze the main issues that must be faced by learning robots and propose a set of techniques aimed at making the RL approach more effective in real robotic tasks. In particular, our main contributions are the introduction of the *lower bound* update strategy, which allows to learn robust policies without the need of a complete exploration of the whole state-action space, and the use of *piecewise-constant policies* with reward accumulation, which allows to efficiently learn even in presence of coarse discretizations of the state space. Experimental results carried out with a real robot show that the proposed learning techniques are effective in making the learning process more stable than traditional RL algorithms.

In the next section, we will briefly review the main approaches proposed in literature to overcome the problems described above. In Section 3, we introduce the RL framework and present the details of our

algorithm. The results of the experimental validation, carried out on a real robot with a soccer task, are reported in section 4. We draw conclusions and propose future research directions in section 5.

2 REINFORCEMENT LEARNING ON ROBOTS

Given the complexity of the development of robotic applications, the possibility to exploit learning techniques is really appealing. In particular, the reinforcement learning research field provided a number of algorithms that allow an agent to learn to behave optimally by direct interaction with the environment without any a priori information.

Up to now, excluding a few notable exceptions, the application of the RL approach has been successful only in small gridworlds and simple simulated control problems. The success obtained in more complex domains (Tesauro, 1995; Sutton, 1996) is mainly due to ad-hoc solutions and the exploitation of domain dependent information.

Several difficulties prevent the use of pure RL methods in real world robotic applications. Since direct experience is the main source of information for an RL algorithm, the learning agent needs to repeatedly interact with the world executing each available action in every state. While in software domains it is possible to perform a large number of trials and to place the agent in arbitrary states, in real robotic domains there are several factors (limited battery capacity, blocking states, mechanical or electrical faults, etc.) that make learning from scratch not feasible.

Several works in the robotic area have studied different solutions to reduce the amount of direct experience required by RL algorithms. A typical solution consists of performing extensive training sessions using a physical simulator (Morimoto and Doya, 2000). When the simulated robot achieves a good performance the learned policy is applied on the physical robot and the learning process goes on with the aim of adjusting it to the real conditions. Although this approach can be really effective, for many robotic domains it is too hard to realize good enough simulation environments. Furthermore, it may happen that the approximation introduced in the simulation is such that the knowledge gathered in the simulated learning phase is almost useless for the real robot. Another approach that was originally proposed for robotic tasks, but that has found common application in other domains, is experience replay (Lin, 1992). The idea is that the robot stores data about states, actions, and rewards experienced, and fictitiously repeats the same moves thus performing more updates that speed up the propagation of the rewards and the convergence to the optimal value function. Although this approach succeeds in speeding up the learning process, it still requires an expensive exploration phase to gather enough information. To overcome this problem, Lin adopts a human teacher to show the robot several instances of reactive sequences that achieve the task in order to bias the exploration to promising regions of the action space. The same goal is pursued in (Millán, 1996), but instead of a teacher it requires a set of pre-programmed behaviors to focus the exploration on promising parts of the action space when the robot faces new situations. In this paper, we follow the approach proposed by (Smart and Kaelbling, 2002), which effectively provides prior knowledge by splitting the learning process in two phases. In the first phase, example trajectories are supplied to the robot (by automatic control or by human guidance) through a control policy and the RL system passively watches the experienced states, actions and rewards with the aim of bootstrapping information into the value-function. Once enough data has been collected, the second phase starts and the robot is completely controlled by the RL algorithm. In problems with sparse reward functions, without any hint, the robot would take a huge number of steps before collecting some significant reward, thus making the learning process prohibitive. At the opposite, the "supervised" phase is an initialization of the learning process so that it can initially avoid a fully random exploration of the environment. Furthermore, differently from imitation learning methods, this approach allows to supply prior knowledge without knowing anything about inverse kinematics.

Learning in real-world environments requires to deal with dangerous actions that may harm the robot or humans, and with stalling states, i.e., configurations that prevent the robot from autonomously going on with the learning process. Again, example trajectories can be effective to provide safe policies that avoid harmful situations. Another way to reduce the risk of performing dangerous actions is to use minimax learning (like the $\widehat{Q}$-learning algorithm (Heger, 1994)), where the robot, instead of maximizing the expected sum of discounted rewards, tries to maximize the value of the worst case. This kind of pessimistic learning has lead to good results in stochastic (Heger, 1994), partially observable (Buffet and Aberdeen, 2006), and multi-agent (Littman, 1994) problems, showing also to be robust with respect to changes in the problem parametrization, thus allowing the reuse of the learned policy in different operating conditions. Unfortunately, algorithms like $\widehat{Q}$-

learning need to perform an exhaustive search through the action space, but this is not feasible in a real-world robotic context. In the following sections, we propose a variant to the $\widehat{Q}$-learning that is able to find safe policies without requiring the complete exploration of the action space.

Another relevant issue in real-world robotic applications is that both the state and action spaces are continuous. Usually, this problem is faced by using function approximators such as state aggregation, CMAC (Sutton, 1996) or neural networks, in order to approximate the value function over the state space. Although these techniques obtained relevant results in supervised learning, they require long hand tuning and may result in highly unstable learning processes and even divergence. Although state aggregation is one of the most stable function approximator, its performance of state aggregation is strictly related to the width of the aggregated states and algorithms like Q-learning may have very poor performance even in simple continuous problems, unless a very fine discretization of the state space is used. In this paper, we propose a learning technique that allows to achieve good policies even in presence of large state aggregations, thus exploiting their generalization properties to reduce the learning times.

3 THE ALGORITHM

As discussed in previous sections, learning in noisy continuous state spaces is a difficult task for many different reasons. In this section, after the introduction of the formal description of the RL framework, we detail a novel algorithm based on the idea of the computation of a lower bound for a piecewise constant policy.

3.1 The Reinforcement Learning Framework

RL algorithms deal with the problem of learning how to behave in order to maximize a reinforcement signal by a direct interaction with a stochastic environment. Usually, the environment is formalized as a finite state discrete Markov Decision Process (MDP):

1. A set of states $\mathcal{S} = \{s_1, s_2, \cdots s_N\}$
2. A set of actions $\mathcal{A} = \{a_1, a_2, \cdots a_M\}$
3. A transition model $\mathcal{P}(s, a, s')$ that gives the probability to get to state s' from state s by taking action a
4. A reward function $\mathcal{R}(s, a)$ that gives the value of taking action a in state s

Furthermore, each MDP satisfies the Markov property:

$$\mathcal{P}(s_{t+1}|s_t, a_t) = \mathcal{P}(s_{t+1}|s_t, a_t, s_{t-1}, a_{t-1}, \cdots, s_0, a_0), \quad (1)$$

that is, the probability of getting in state s at time $t+1$ depends only on the state and action at the previous time step and not on the history of the system.

A deterministic policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$ is a function that maps each state in the environment to the action to be executed by the agent. The action value function $Q^\pi(s)$ measures the utility of taking action a in state s and following a policy π thereafter:

$$Q^\pi(s,a) = R(s,a) + E\left[\sum_{t=1}^{\infty} \gamma^t R(s(t), \pi(s(t)))\right], \quad (2)$$

where $\gamma \in [0,1)$ is a discount factor that weights recent rewards more than those in the future.

The goal of the agent is to learn the optimal policy π^* that maximizes the expected discounted reward in each state. The action value function corresponding to the optimal policy can be computed by solving the following Bellman equation (Bellman, 1957):

$$Q^*(s,a) = R(s,a) + \gamma \sum_{s'} \mathcal{P}(s,a,s') \max_{a'} Q^*(s',a'). \quad (3)$$

Thus, the optimal policy can be defined as the greedy action in each state:

$$\pi^*(s) = arg\max_a Q^*(s,a). \quad (4)$$

Q-learning (Watkins and Dayan, 1992) is a model-free algorithm that incrementally approximates the solution through a direct interaction with the environment. At each time step, the action value function $Q(s,a)$ is updated according to the reward received by the agent and to the estimation of the future expected reward:

$$Q^{k+1}(s,a) = (1-\alpha)Q^k(s,a) + \alpha\left[r + \gamma \max_{a'} Q^k(s',a')\right],$$

where $r = R(s,a)$ and α is the learning rate. In the following, we will refer to the term in square brackets as the *target value*:

$$U(s,a,s') = R(s,a) + \gamma \max_{a'} Q^k(s',a') \quad (5)$$

When α decreases to 0 according to the Robbins-Monro (Sutton and Barto, 1998) conditions and each state-action pair is visited infinitely often, the algorithm is proved to converge to the optimal action value function. Usually, at each time step the action to be executed is chosen according to an explorative policy that balances a wide exploration of the environment and the exploitation of the learned policy. One of the most used exploration policies is ε-greedy (Sutton, 1996) that chooses the greedy action with probability $1-\varepsilon$ and a random action with probability ε.

3.2 Local Exploration Strategy

In order to avoid an exhaustive exploration of the action space, a more sophisticated exploration policy can be adopted. As suggested in (Smart and Kaelbling, 2002), a "supervised" phase performed using a sub-optimal controller is an effective way to initialize the action value function. This way, the learning process is bootstrapped by a hand-coded controller whose policy is optimized when the control of the robot is passed to the learning algorithm. Even if this technique is effective to avoid an initial random exploration, the ε-greedy exploration policy used thereafter does not guarantee that many useless, and potentially dangerous, actions are explored. To reduce this problem, we propose to adopt a *local* ε-greedy exploration policy. Since the learning process is initialized using a sub-optimal controller, it is preferable to perform an exploration in the range of the greedy action, instead of completely random actions. Therefore, with probability ε a locally explorative action is drawn from a uniform probability distribution over a δ-interval of the greedy action a^* [1]:

$$a_{EXP} \sim U(a^* - \delta; a^* + \delta) \tag{6}$$

Even if the *local* ε-greedy exploration policy is not guaranteed to avoid dangerous explorative actions, in many robotic applications it is likely to be safer and to converge to the optimal policy in less learning episodes than usual ε-greedy policy. In fact, it is based on the assumption (often verified in robotic applications) that the optimal policy can be obtained by small changes to the sub-optimal controller.

3.3 Q_{LB}-learning

As showed in many works (e.g., (Gaskett, 2003; Morimoto and Doya, 2001)) traditional Q-learning is often ill-suited for robotic applications characterized by noisy continuous environments with several uncertain parameters in which non-stationary transitions may occur because of external unpredictable factors. Many techniques (Gaskett, 2003; Morimoto and Doya, 2001; Heger, 1994) improve the stability and the robustness of the learning process, on the basis of the concept of maximization of performance in the worst case, i.e., the *min-max* principle. This principle deals with the problems introduced by highly uncertain and stochastic environments. In fact, the controller obtained at the end of the learning process is optimized for the worst condition and not for the average situation, such as in Q-learning. Although effective in principle, this approach cannot always be applied in real world applications. The *Robust Reinforcement Learning* (RRL) paradigm (Morimoto and Doya, 2001) relies on an estimation of the dynamics and of the noise of the environment in order to compute the min-max solution of the value function. Unfortunately, the model of the environment is not always available and the estimation of its dynamics often requires many learning episodes. A model-free solution, the $\widehat{Q}$-learning algorithm, proposed in (Heger, 1994) can learn a robust controller through a direct interaction with the environment. The update formula for the action value function is:

$$Q^{k+1}(s,a) = \min\left[Q^k(s,a), U(s,a,s')\right], \tag{7}$$

where $U(s,a,s')$ is the target value. If the action value function is initialized to the highest possible value (i.e., $Q(s,a) = \frac{R_{max}}{1-\gamma}$), this algorithm is proved to converge to the min-max value function and policy, that is the policy that receives the highest expected reward in the worst case. Although this algorithm is guaranteed to find a robust controller, it can be applied only to simulated environments, since the optimistic initialization of the action value function makes the agent to explore randomly all the available actions until at least the best action converged to the min-max value function. Thus, it is not suitable for robotic applications where long exploration is too expensive.

Another drawback of $\widehat{Q}$-learning is that it finds an optimal policy for the worst case even if caused by non-stationary transitions. In fact, in real-world applications very negative conditions may occur during the learning process because of very limited and uncontrolled situations possibly caused by non-stationarity in the environment and, with $\widehat{Q}$-learning, these conditions are immediately stored in the action value function and cannot be removed anymore.

In order to keep the robustness of a min-max controller, to reduce the exploration and to avoid effects of non-stationarity as much as possible, we propose Q_{LB}-learning, a novel algorithm for the computation of a lower bound for the action value function. Instead of a minimization between the current estimation and the target value (Eq. 5), we adopt the following update rule:

$$Q^{k+1}(s,a) = \begin{cases} U(s,a,s') & \text{if } U(s,a,s') < Q^k(s,a) \\ (1-\alpha)Q^k(s,a) + \alpha U(s,a,s') & \text{otherwise} \end{cases}$$

As it can be noticed, when the worst case is visited the action value estimation is set to the target value as in $\widehat{Q}$-learning. On the other hand, if the target received

[1] In case of problems with multiple actuators, the explorative action is obtained by the composition of explorative actions for each actuator.

by the agent is greater than the current estimation, the usual Q-learning update rule is used. As a result, the action value function may not take into consideration the worst case ever visited in the learning process, when this is the result of rare events not following the real dynamics of the system (e.g., collisions against moving obstacles). As learning progresses the learning rate α decreases (according to Robbins-Monro conditions) thus granting the convergence of the Q_{LB}-learning algorithm since it becomes more and more similar to $\widehat{Q}$-learning [2]. As it can be noticed, this algorithm does not require any particular initialization of the action value function as in $\widehat{Q}$-learning and this can reduce the exploration needed to learn a nearly optimal solution. Furthermore, this algorithm is effective in case of continuous state spaces in which the transitions between states may be affected also by the policy the robot is performing (Moore and Atkeson, 1995). In this situation, the worst case depends on the policy and not only on the dynamics of the environment, thus it is necessary to evaluate the action value function according to the current policy and not with respect to the worst possible case. Therefore, while $\widehat{Q}$-learning would converge to the worst case independently from the policy, Q_{LB}-learning learns the action value function for the worst case of the current policy.

3.4 PWC-Q-learning

Although the previous algorithm is effective in noisy or non-stationary environments, it may experience bad results (see Section 4) when applied to problems with continuous state spaces. Usually, when applying RL algorithms, a continuous state space is discretized into intervals that are considered as aggregated states. Unfortunately, if a coarse discretization is adopted, the environment looses the Markov property and the learning process is likely to fail. In fact, a very fine resolution on the state space is required to make algorithms such as Q-learning stable and effective. Since the number of states has a strong impact on the speed of the learning process, this is often a problem of its application to robotic problems.

The reason for Q-learning to fail in learning on coarsely discretized states is strictly related to its learning process, that continuously performs updates within the same state. With Q-learning, when the robot takes the greedy action a in state s, receives a reward r and remains in the very same state, the action value function is updated as:

$$Q^{k+1}(s,a) = (1-\alpha)Q^k(s,a) + \alpha\left[r + \gamma Q^k(s,a)\right] \quad (8)$$

[2] Let us notice that when $\alpha = 0$ Q_{LB}-learning is the same as the $\widehat{Q}$-learning

Thus, the value of $Q(s,a)$ is updated using its own estimation. If the state is sufficiently large, the agent is likely to remain in the same state for many steps and the Q-value tends to converge to its limit $\frac{r}{1-\gamma}$ until either the state is left or another action is chosen. As a result, the policy continuously changes and the learning process may experience instability. This phenomenon is much more relevant when a *min-max*-based update rule is adopted; in that case, the convergence to the limit is even faster.

In order to avoid the negative effect of the self-update rule, we introduce a novel learning algorithm: the *Piecewise Constant Q-learning* (PWC-Q-learning). The main difference with respect to the traditional Q-learning is about the way the action value function is updated. When the robot enters a state s and selects an action a, PWC-Q-learning makes the robot repeatedly execute the same action a and it accumulates the reward until a state transition occurs. Only at that time, the update is performed according to the SMDP Q-learning rule (Sutton et al., 1999):

$$\begin{aligned} Q^{k+1}(s,a) &= (1-\alpha)Q^k(s,a) + \\ &\quad \alpha\left[\sum_{i=1}^{N}\gamma^{i-1}r_i + \gamma^N \max_{a'} Q^k(s',a')\right] \end{aligned}$$

where N is the number of times in which the agent has performed action selection in the state s. As it can be noticed, in this way s' is always different from s and no self-update is performed. As a result, the PWC-Q-learning is more stable and guarantees a more reliable learning process than the original Q-learning.

Furthermore, the PWC-Q-learning algorithm matches also the limitations caused by the discretization on the resolution of the controller that can be actually learned. In fact, when the learning is over, the learned policy π maps each state into one single action that must be kept constant until a different state is perceived. Therefore, in this case a learning algorithm as PWC-Q-learning that evaluates the real utility of an action throughout a state is more suitable, and does not allow any change in the action as in Q-learning.

While Q-learning needs a very fine discretization to reduce the instability caused by the loss of the Markov property, PWC-Q-learning (as shown in the experimental section) proved to be more stable even in coarsely discretized continuous state spaces. By using coarse discretizations it is also possible to reduce the duration of the learning process.

Finally, PWC-Q-learning can be merged with the computation of the lower bound action value function introduced in Section 3.3 in order to obtain a learning algorithm that is robust in highly stochastic and noisy environments and that, at the same time, can be successfully applied to continuous robotic problems.

Figure 1: The RoboCup robot used for the experiments.

4 EXPERIMENTAL RESULTS

To verify the effectiveness of the proposed approach we made real robotic experiments with the aim of measuring speed and stability of the learning process, and optimality and robustness of the learned policy. The experimental activity has been performed on a real robot belonging to the Milan RoboCup Team (MRT) (Bonarini et al., 2006), a team of soccer robots that participates to the Middle Size League of the RoboCup competition (Kitano et al., 1997). The robot used for the experiments is a holonomic robot with three omnidirectional wheels that can reach the maximum speed of $1.8m/s$ (see Figure 1). The robot is equipped with a omnidirectional catadioptric vision sensor able to detect objects (e.g., ball, robots) up to a distance of about $6m$ all around the robot.

Preliminary experiments have been carried out on the task "go to ball", in which the robot must learn how to reach the ball as fast as possible. Although we have chosen a quite simple task, the large discretization adopted, the non-stationarity of the environment (due to battery consumption during the learning process), the noise affecting robot's sensors and actuators, and the limited amount of experience make the results obtained significant to evaluate the benefits of the PWC-Q_{LB}-learning algorithm. The state space of this task is characterized by two continuous state variables: the *distance* and the *angle* at which the robot perceives the ball. The ball distance has been discretized into five intervals: $[0:50), [50:100), [100:200), [200:350), [350:600]$, while the angle has been evenly split into 24 sectors 15° wide, thus obtaining a state space with 120 states. As far as the action space is concerned, thanks to its three omnidirectional wheels, the robot can move on the plane with three degrees of freedom mapped to three action variables:

- *module of the tangential velocity*, associated to the speed at which the robot translates. Its value is expressed in percentage of the maximum tangential velocity and it is discretized into 6 values $0, 20, 40, 60, 80, 100$;
- *direction of the tangential velocity*, the direction along with the robot translates. Its value is expressed in degrees and it is discretized into 24 evenly spaced values: $0^\circ, 15^\circ, 30^\circ, \ldots, 345^\circ$;
- *rotational velocity*, associated to the speed at which the robot changes its heading. Its value is expressed in percentage of the maximum rotational velocity and it is discretized into 9 values: $-20, -15, -10, -5, 0, 5, 10, 15, 20$.

The total number of available actions is $1,296$.

The reward function is such that the robot receives -1 as reward at each step except when its distance from the ball is below $50cm$ and the angle falls in the range $[-15^\circ : 15^\circ]$, in which case the reward is $+10$ and the trial ends. At the beginning of each learning trial, the robot starts form the center kickoff position and performs learning steps until it succeeds in reaching the ball which is positioned at $350cm$. Once a trial is finished, the learning process is suspended, and the robot autonomously performs a resetting procedure moving towards the starting position.

The sparsity of this reward function, although making easier its definition and preventing the introduction of biases, requires a long exploration period before catching some positive rewards and propagate the associated information to the rest of the state space. In this task, the robot would clueless wander around the field with little hope of success and high risk of bumping against objects around the field. For this reason, as mentioned in Section 2, we split the learning process into two phases. The first phase consists of "supervised" trials, i.e., trials in which the robot is controlled by hand-written behaviors and the RL algorithm only observes and records the actions taken in the visited states and the associated rewards. On the basis of the observed data, the RL algorithm builds a first approximation of the value function that will be exploited in the second phase. The acquired policy allows to make a safe exploration of the environment, thus considerably speeding up the learning process towards the optimal policy. In the second phase, the hand-written controller is bypassed by the RL system which chooses which action must be executed by the robot in each state, and, on the basis of the collected reward and the reached state, performs on-line updates of its knowledge.

In the following, we present comparative experiments among Q-learning, Q_{LB}-learning (see Section 3.3), and PWC-Q_{LB}-learning (see Section 3.4). The "supervised" phase has been performed only once and then the collected data have been reused in all the experiments. It consists of 60 trials with the ball placed in different positions around the robot within $400cm$. Since the actions produced by the

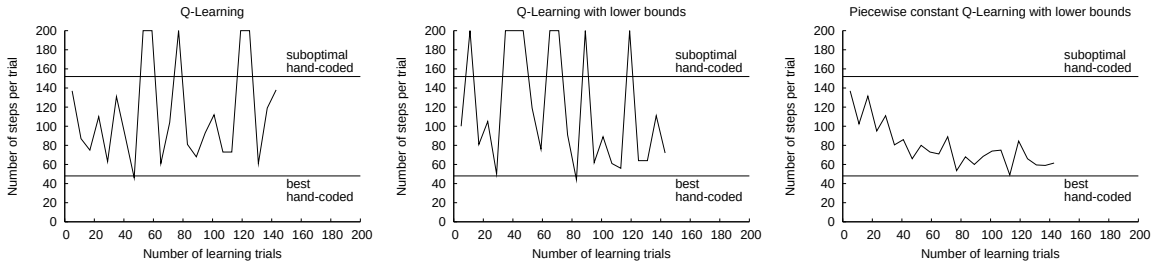


Figure 2: Performance of Q-learning, Q_{LB}-learning and PWC-Q_{LB}-learning on the "go to ball" task.

hand-written controller (based on fuzzy rules) are continuous, and given that the RL algorithms work with discrete actions, we replace each action produced by the controller with the one with the closest value among those in the discrete set of the RL system. Given the low complexity of the task, the hand-written behavior is, on purpose, highly suboptimal (with only low speed commands) in order to better highlight the improvements obtained by the learning processes in such a noisy environment.

In Figure 2 are reported the learning plots of the three algorithms during the second learning phase. The performance of the algorithms is measured by the number of steps (each step lasts 70*ms*) required to reach the ball. Every five trials the learning process is suspended for one trial; in this trial the robot executes the policy learned so far. Only the exploitation trials are shown in the graphs. When the robot is not able to reach the ball within 200 steps, the trial ends. For each graph are also reported the average performance of the suboptimal hand-coded policy followed by the controller ($\sim$ 152 steps) and the average performance of our best hand-coded policy ($\sim$ 48 steps). It is worth noting that, due to noise in the sensing and actuating systems and to the low accuracy of the resetting procedure, also the performance of fixed policies is affected by a high variance (standard deviation of about $\pm$10 steps). The parametrization is the same for each learning algorithm: the learning rate is 0.5 and decreases quite quickly, the ε-greedy exploration starts at 0.5 and decreases slowly, the discount factor is 0.99. The action values stored in the Q-table are initialized with a low value (-100), so that the values of the actions performed during the supervised trials become larger, thus biasing the exploration to regions near to the example trajectories. Given the stochasticity (due to sensor noise) and the non-stationarity (due to battery discharging), each experiment has been repeated three times for each algorithm, for a total of $1,350$ trials and more than $130,000$ steps.

In the first two graphs we have reported typical runs of Q-learning and Q_{LB}-learning. Both of them show a quick learning in the first trials, but then they alternate good trials to bad trials without appearing to converge to a stable solution. Trials that reach 200 steps typically mean that the robot, in some regions, has learned a policy that stops the movement of the robot. As explained above, this irrational behavior is due to self-updates (see Equation 8). Unsurprisingly, this problem is more frequent in the Q_{LB}-learning algorithm. The third plot displays the performance of PWC-Q_{LB}-learning algorithm averaged over three different runs. Here, the number of steps needed to reach the ball decreases quite slowly, but, unlike the other algorithms, the policy improves more and more reaching performance close to the best hand-coded policy, without any trial reaching the 200 steps limit.

It is worth briefly describing the policy learned by the robot with the PWC-Q_{LB}-learning algorithm: in the first trials the robot gradually learns to increase its speed in different situations until it learns to reach the ball at the maximum speed. Although this policy allows to complete some trials in very few steps (even less than 40 steps), it is likely to fail, since a coarse discretization gives control problems at high speed, especially in regions where a good accuracy is required. The problem is that, when the robot travels fast, it may not be able to go straight to the ball, and in general it hits the ball, thus needing to run after it for many steps. Given the risk aversion typical of minimax approaches, the PWC-Q_{LB}-learning algorithm quickly learns to give up with fast movements when the robot is near to the ball. The final result is that the robot starts at high speed and slows down when the ball gets closer, thus managing to reach the termination condition without touching the ball.

During one of the experiments another interesting characteristic of the PWC-Q_{LB}-learning algorithm emerged. After about 100 trials, one of the wheels began to partially lose its grip on the engine axis, thus causing the robot to turn slightly right instead of moving straightforward. Our learning algorithm managed

to face this non-stationarity by reducing the tangential speed in all the states, turning slightly right, and then it started to gradually raise the tangential speed. This behavior has put in evidence how this learning approach potentially can increase the robustness towards mechanical faults, thus increasing the autonomy of the robot. Further studies will be focused on a detailed analysis of this important aspect.

5 DISCUSSION AND FUTURE WORKS

The application of RL techniques to robotic applications could lead to the autonomous learning of optimal solutions for many different tasks. Unfortunately, most of the traditional RL algorithms fail when applied to real-world problems because of the time required to find the optimal solution, of the noise that affects both sensors and actuators, and of the difficulty to manage continuous state spaces.

In this paper, we have described and experimentally tested a novel algorithm (PWC-Q_{LB}-learning) designed to overcome the main issues that arise when learning is applied to real-world robotic tasks. PWC-Q_{LB}-learning computes the lower bound for the action value function while following a piecewise constant policy. Unlike other *min-max*-based algorithms, PWC-Q_{LB}-learning does not require the model of the dynamics of the environment and avoids long and blind exploration phases. Furthermore, it does not learn the optimal policy for the theoretically worst case, but it estimates the lower bound on the conditions actually experienced by the robot according to its current policy and to the current dynamics of the environment. Finally, the piecewise constant action selection and update guarantee a stable learning process in continuous state spaces, even when the discretization is such that the Markov property is lost.

Although preliminary, the experiments showed that PWC-Q_{LB}-learning succeeds in learning a nearly-optimal policy by optimizing the behavior of a sub-optimal controller in noisy continuous environments. Furthermore, it proved to be more stable with respect to Q-learning even when a coarse discretization of the state space is used. At the moment, we are currently investigating the theoretical properties of the proposed algorithm and we are testing its performance on more complex robotic tasks, such as the "align to goal" and the "kick" tasks.

REFERENCES

Bellman, R. (1957). *Dynamic Programming*. Princeton University Press, Princeton.

Bonarini, A., Matteucci, M., Restelli, M., and Sorrenti, D. G. (2006). Milan robocup team 2006. In *RoboCup-2006: Robot Soccer World Cup X*.

Buffet, O. and Aberdeen, D. (2006). Policy-gradient for robust planning. In *Proceedings of the Workshop on Planning, Learning and Monitoring with Uncertainty and Dynamic Worlds (ECAI 2006)*.

Gaskett, C. (2003). Reinforcement learning under circumstances beyond its control. In *Proceedings of International Conference on Computational Intelligence for Modelling Control and Automation*.

Heger, M. (1994). Consideration of risk in reinforcement learning. In *Proceedings of the 11th ICML*, pages 105–111.

Kitano, H., Asada, M., Osawa, E., Noda, I., Kuniyoshi, Y., and Matsubara, H. (1997). Robocup: The robot world cup initiative. In *Proceedings of the First International Conference on Autonomous Agent (Agent-97)*.

Lin, L.-J. (1992). Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine Learning*, 8(3-4):293–321.

Littman, M. L. (1994). Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the 11th ICML*, pages 157–163.

Millán, J. D. R. (1996). Rapid, safe, and incremental learning of navigation strategies. *IEEE Transactions on Systems, Man, and Cybernetics (B)*, 26(3):408–420.

Moore, A. and Atkeson, C. (1995). The parti-game algorithm for variable resolution reinforcement learning in multidimensional state-spaces. *Machine Learning*, 21:711–718.

Morimoto, J. and Doya, K. (2000). Acquisition of stand-up behavior by a real robot using hierarchical reinforcement learning. In *Proceedings of the 17th ICML*.

Morimoto, J. and Doya, K. (2001). Robust reinforcement learning. In *Advances in Neural Information Processing Systems 13*, pages 1061–1067.

Smart, W. D. and Kaelbling, L. P. (2002). Effective reinforcement learning for mobile robots. In *Proceedings of ICRA*, pages 3404–3410.

Sutton, R. S. (1996). Generalization in reinfrocement learning: Successful examples using sparse coarse coding. In *Advances in Neural Information Processing Systems 8*, pages 1038–1044.

Sutton, R. S. and Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA.

Sutton, R. S., Precup, D., and Singh, S. (1999). Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211.

Tesauro, G. (1995). Temporal difference learning and td-gammon. *Communications of the ACM*, 38.

Watkins, C. and Dayan, P. (1992). Q-learning. *Machine Learning*, 8:279–292.

DISCRETE DYNAMIC SLIDING SURFACE CONTROL FOR ROBUST SPEED CONTROL OF INDUCTION MACHINE DRIVE

Abdel Faqir[(1)], Daniel Pinchon[(2)], Rafiou Ramanou[(2)] and Sofiane Mahieddine[(2)]
[(1)] ECTEI – (Paris, France)
faqir@ece.fr
[(2)] LTI - Institut Universitaire de Technologie de l'Aisne (GEII)
13, Avenue François Mitterrand
02880 Cuffies
France
Tel: +33 (0) 323 764 010, Fax: +33 (0) 323 764 015
daniel.pinchon@u-picardie.fr

Keywords: Induction machine, indirect field oriented control, sliding mode control.

Abstract: This paper proposes the discrete dynamic sliding surface control to guarantee the existence of discrete sliding mode and reduce the chattering phenomena for speed control of induction machine drive. In discrete systems, the controller does not control the system during the sampling interval. The great chattering and large control signal are caused by the high switching gain. In this paper, the dynamic sliding surface is introduced to overcome the drawback. By setting the initial value of the dynamic sliding surface, the system can lock to the sliding surface quickly without high switching gain. The control signal can be reduced and the chattering can be eliminated. Furthermore, the induction machine speed control system is used to show this controller's robustness to against the parameter variation and external load. The speed of the induction machine is regulated using the indirect field oriented control (IFOC). Thus, after the application of the IFOC technique by determining the decoupled model of the machine, a discrete sliding surface controller has been applied. Simulation study is used to show the performances of the proposed method and then validated by an experimental prototype.

1 INTRODUCTION

Field oriented control, published for the first time by Blaschke in his pioneering work in 1972, consists in adjusting the flux by a component of the current and the torque by the other component. For this purpose, it is necessary to choose a d-q reference frame rotating synchronously with the rotor flux space vector, in order to achieve decoupling control between the flux and the produced torque. This technique allows to obtain a dynamical model similar to the DC machine.

This technique presents a major drawback. Indeed the behavior of the machine and its command is strongly affected by the variation of the rotor resistance due to the temperature or by the variations of the rotor inductance due to the saturation.

To eliminate this drawback, we propose in this paper, an indirect field oriented method using two sliding mode controllers. Once the decoupled model of the machine is obtained, a discrete sliding surface control is chosen with an appropriate switching. Simulations have been carried out to verify the effectiveness and the performances of the proposed method.

2 SYSTEM DESCRIPTION AND MACHINE MODELLING

The system is an induction machine fed by a PWM voltage source inverter. The sliding mode controllers are applied to the inverter via reference voltages (Fig. 1).

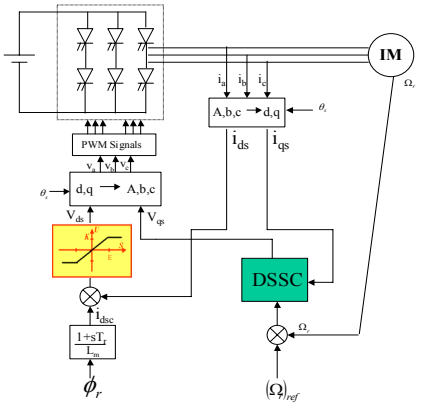


Figure 1: Discrete Sliding Surface Control structure of an induction machine.

2.1 Induction Machine Model

For the study of the induction machine, we take the following model:

$$\frac{dX}{dt} = AX + BU \tag{1}$$

with

$$A = \begin{bmatrix} -\left(\frac{1}{T_s\sigma}+\frac{1}{T_r}\frac{1-\sigma}{\sigma}\right) & \omega_s & \frac{1-\sigma}{\sigma}\frac{1}{L_m T_r} & \frac{1-\sigma}{\sigma}\frac{1}{L_m}\omega_r \\ -\omega_s & -\left(\frac{1}{T_s\sigma}+\frac{1}{T_r}\frac{1-\sigma}{\sigma}\right) & -\frac{1-\sigma}{\sigma}\frac{1}{L_m}\omega_r & \frac{1-\sigma}{\sigma}\frac{1}{L_m T_r} \\ \frac{L_m}{T_r} & 0 & -\frac{1}{T_r} & \omega_{sl} \\ 0 & \frac{L_m}{T_r} & -\omega_{sl} & -\frac{1}{T_r} \end{bmatrix}$$

$$B = \begin{bmatrix} \frac{1}{\sigma L_s} & 0 \\ 0 & \frac{1}{\sigma L_s} \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad X = \begin{bmatrix} i_{ds} \\ i_{qs} \\ \phi_{dr} \\ \phi_{qr} \end{bmatrix} \text{ and } U = \begin{bmatrix} v_{ds} \\ v_{qs} \end{bmatrix}$$

Where

$$\sigma = \left(1-\frac{L_m^2}{L_s L_r}\right) \qquad \omega_r = p\Omega_r, \qquad \omega_s = \omega_{sl} + \omega_r.$$

The stator voltages (v_{ds}, v_{qs}) are considered as control inputs, while the stator currents (i_{ds}, i_{qs}), the rotor flux (ϕ_{dr}, ϕ_{qr}) and the speed (Ω_r) are considered as state variables.

From the equations (1), the following electrical equations are deduced:

$$\frac{di_{ds}}{dt} = \frac{-R_{sm}}{\sigma L_s} i_{ds} + \omega_s i_{qs} + \frac{1}{\sigma L_s}\frac{L_m}{T_r L_r}\phi_{dr} + \frac{1}{\sigma L_s}\frac{L_m}{L_r}\omega_r\phi_{qr} + \frac{1}{\sigma L_s}v_{ds} \tag{2}$$

$$\frac{di_{qs}}{dt} = -\omega_s i_{ds} - \frac{R_{sm}}{\sigma L_s} i_{qs} - \frac{1}{\sigma L_s}\frac{L_m}{L_r}\omega_r\phi_{dr} + \frac{1}{\sigma L_s}\frac{L_m}{T_r L_r}\phi_{qr} + \frac{1}{\sigma L_s}v_{qs} \tag{3}$$

$$\frac{d\phi_{dr}}{dt} = \frac{L_m}{T_r} i_{ds} - \frac{1}{T_r}\phi_{dr} + \omega_{sl}\phi_{qr} \tag{4}$$

$$\frac{d\phi_{qr}}{dt} = \frac{L_m}{T_r} i_{qs} - \omega_{sl}\phi_{dr} - \frac{1}{T_r}\phi_{qr} \tag{5}$$

With $R_{sm} = R_s + \frac{L_m^2}{L_r^2} R_r$

The mechanical model is given by:

$$J\frac{d\Omega_r}{dt} = T_{em} - T_L - K_f\Omega_r \tag{6}$$

And the electromagnetic torque can be expressed as:

$$T_{em} = \frac{pL_m}{L_r}\left(\phi_{dr} i_{qs} - \phi_{qr} i_{ds}\right) \tag{7}$$

2.2 Field Oriented Control

The field orientation is obtained by imposing:

$$\begin{cases} \phi_{dr} = \phi_r \\ \phi_{qr} = 0 \end{cases} \tag{8}$$

From the equations (4) and (8), the i_{ds} reference can be computed in order to impose the flux ϕ_r :

$$\phi_r = \frac{L_m}{1+sT_r} i_{ds} \tag{9}$$

Furthermore, the position θ_s of the rotating frame can be estimated using equations (5) and (8):

$$\theta_s = \int\left(\frac{L_m i_{qs}}{T_r\phi_r} + p\Omega_r\right)dt \tag{10}$$

With taking into account the field orientation of the machine, the stator equations on d-q axis become:

$$\begin{cases} V_{ds} = \left[\sigma L_s\frac{di_{ds}}{dt} + R_{sm} i_{ds} - \sigma L_s\omega_s i_{qs} - \frac{L_m}{T_r L_r}\phi_r\right] \\ V_{qs} = \left[\sigma L_s\frac{di_{qs}}{dt} + R_{sm} i_{qs} + \sigma L_s\omega_s i_{ds} + \frac{L_m}{L_r}\omega_r\phi_r\right] \end{cases} \tag{11}$$

2.3 Decoupling System

Using the system given by equations (11), we can remark the interaction of both inputs, which makes the control design more difficult.

The first step of our work is to obtain a decoupled system in order to control the electromagnetic torque via stator quadrature current i_{qs} such as a DC machine.

A decoupled model can be obtained by using two intermediate variables:

$$v_{ds1} = v_{ds} + emf_d \tag{12}$$

$$v_{qs1} = v_{qs} + emf_q \tag{13}$$

where $emf_d = \omega_s\sigma L_s i_{qs} + \frac{L_m}{L_r^2} R_r\phi_r$ (14)

and $emf_q = -\omega_s \sigma L_s i_{ds} - \omega_s \frac{L_m}{L_r}\phi_r + \frac{L_m^2}{L_r T_r} i_{qs}$ (15)

$$\begin{bmatrix} v_{ds1} \\ v_{qs1} \end{bmatrix} = M^{-1} \begin{bmatrix} i_{ds} \\ i_{qs} \end{bmatrix}$$

$$M = \begin{bmatrix} L & 0 \\ 0 & L \end{bmatrix} \text{ with } L = \frac{L_r T_r}{R_S L_r T_r + L_m^2 + \sigma L_S L_r T_r s} \quad (16)$$

The stator voltages (v_{ds}, v_{qs}) are reconstituted from (v_{ds1}, v_{qs1}) (Fig. 2):

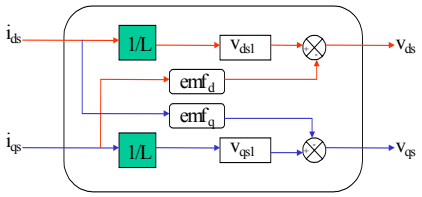


Figure 2: Decoupling control.

3 DISCRETE SLIDING SURFACE CONTROL

3.1 General Concept

Since Dr. Utkin proposed the variable structure system (VSS), it had been widely discussed and applied in many control systems. Due to the change of the switching gains in control function, the controlled system can vary its own controller according to the external condition. Hence, VSS is robust to against to the system's parameter variation and external disturbance. VSS owns one sliding surface predetermined according to the desired dynamic character. Once the sliding mode locking on the sliding surface, the system response will be directed by this surface. The existence condition of classical sliding mode in continuous system is

$$S\dot{S} < 0 \quad (17)$$

Change the differential equation to difference equation. When applying the condition to discrete systems, the existence condition becomes to

$$S(k)[S(k+1) - S(k)] < 0 \quad (18)$$

However, the system is controlled by the controller only in each sampling time. The controller can not modify the response during the sampling interval. It may happen that the condition (18) is not only satisfied but also the sliding motion is divergent. It is shown in fig 3.

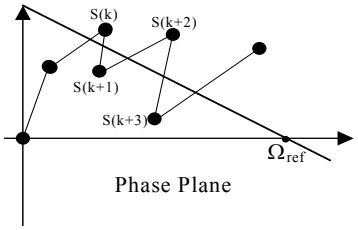


Figure 3: Discrete sliding mode.

The condition (18) only makes the sliding motion toward to the sliding surface. However, it can not guarantee the sliding mode convergent to this surface. The condition (18) is only the necessary condition not the sufficient condition in discrete systems. To make up the drawback, we introduces one additional restriction, that is

$$|S(k+1)| < |S(k)| \quad (19)$$

Combining equation (18) and (19) can make sure the sliding motion convergent. However, the sliding surface is changed to sliding region shown in fig 4.

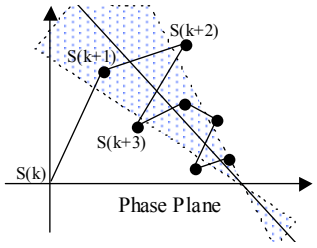


Figure 4: Discrete sliding mode with sliding region.

The choice of switching gain becomes three states. This change causes some difficulty in implementation of hardware. To maintain the binary choice, one restriction of different viewpoint, that is

$$|S(k+1) - S(k)| < \frac{\xi}{2} \quad (20)$$

where ξ is a small positive constant. The varying of each step of sliding motion is restricted. Then, condition (18) makes the sliding mode toward to the surface. The condition (20) makes the sliding motion oscillated on this surface within a small range ξ shown in fig 5.

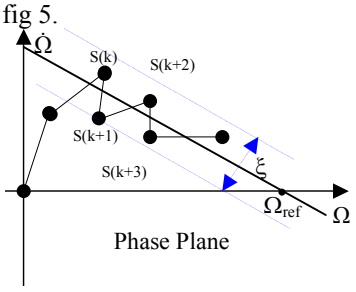


Figure 5: New discrete sliding mode.

This new sliding mode is $|S(k)| < \xi$ different to the classical sliding mode $S(k) = 0$. Hence, this new sliding mode is called "non-ideal sliding mode." This paper will prove that if the controller makes the system's solution of classical sliding mode asymptotically stable, then the same controller makes the solution of non-ideal sliding mode asymptotically stable, too. Hence, the discussion of non-ideal sliding mode can be done like classical sliding mode. Just like the classical sliding mode, the system's dynamic character is directed by the surface only when the sliding motion is within the range of ξ. To reduce the time of out of control, the high switching gain is usually chosen to speed up the reaching time. However, in discrete systems the controller only modifies the control signal at each sampling time. High switching gains can speed up the reaching time, but the chattering often be enlarged. To eliminate the chattering and decrease the reaching time, this paper introduces the dynamic sliding surface control (DSSC) rule shown in fig 6.

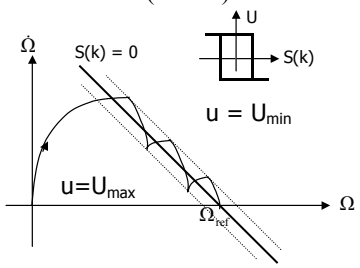


Figure 6: Discrete sliding mode with dynamic sliding surface.

5 SIMULATION RESULTS

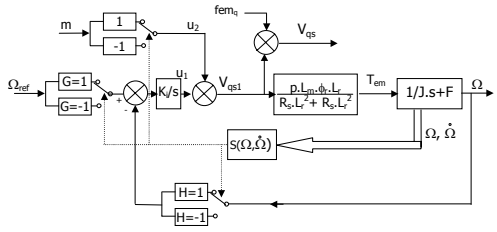


Figure 7: Scheme of speed sliding mode regulation.

The proposed scheme has been simulated using parameters given in the appendix.
The first simulation realized on the AC machine consists in step variation of the reference. Indeed, (Fig. 8) and (Fig. 9) represent the speed and the rotor flux when reference step is first imposed and then an inversion is imposed.

In (Fig. 9) it is clearly shown for rotor flux responses that the decoupling is realized since the direct component of the rotor flux converges to the reference $(\phi_r)_{ref}$, and its quadrature component to zero despite the reference variations. Furthermore we can remark that the proposed control scheme presents good tracking capacities since there is no overshoot and no static error (Fig. 8).

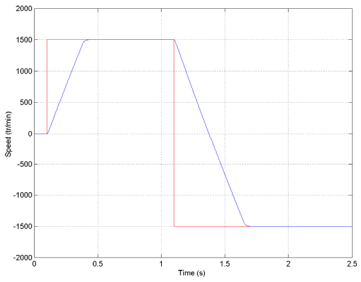


Figure 8: Speed response.

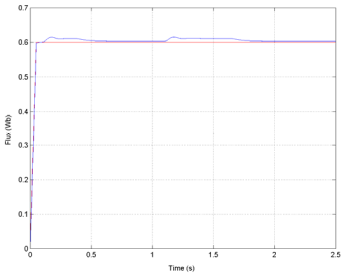


Figure 9: Rotor flux components responses.

(Fig. 10) represents the dynamic response of the speed for different values of the load torque. First when the speed reaches its reference value (1500 tr/min), a step of load torque is applied at (t=0.7 s). The electromagnetic torque rises to the new value of the load torque (Fig.11), and the speed is not disturbed. Then when the load torque is decreased to zero (t=1.2 s) or to a negative value (t=1.5 s) the speed stays on the reference value.
It is clearly shown from the results that the control scheme presents good regulation capacities. Indeed, the external disturbances such as load torque variations are rejected by control system.

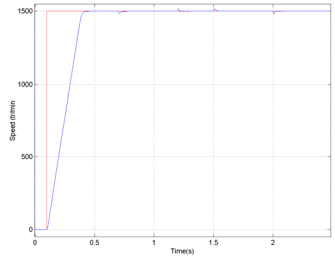


Figure 10: Speed response to load torque variations.

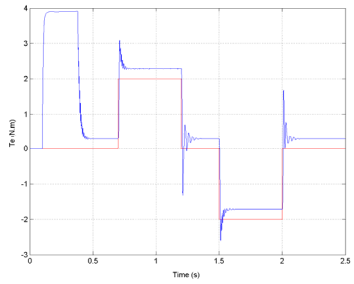


Figure 11: Electromagnetic torque responses to load torque variations.

(Fig. 12) and (Fig. 13) illustrate the dynamic response in the phase plane respectively with a normal sliding surface control and with a dynamic discrete sliding surface control.
We can note the apparition of the chattering phenomenon (Fig. 12) with the normal sliding surface control due to the discontinuous characteristic of this function.

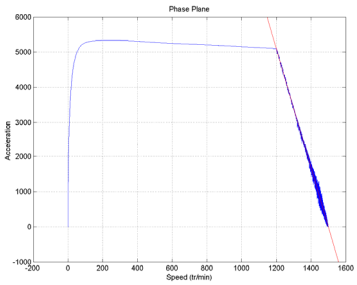


Figure 12: Response in the phase plane with the normal sliding surface control.

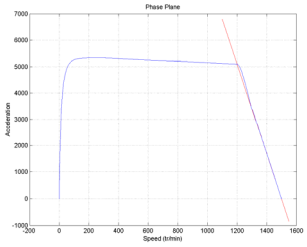


Figure 13: Response in the phase plane with dynamic discrete sliding surface control.

(Fig. 14) depicts the drive response for different values of the rotor constant time.
It is important to note that the changing parameters are introduced only in the model of the machine. The controller is not involved by these variations.
It is well-known for classic controller that the indirect field oriented control is very sensitive to the rotor constant time variations.
The results shown on (Fig. 14) confirm the robustness quality inherent to the proposed controller. Indeed, there is no overshot whatever the rotor constant time

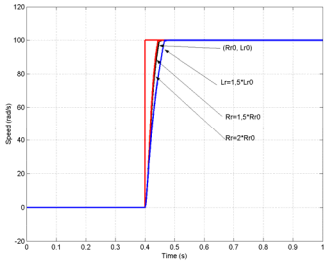


Figure 14: Robustness test of the sliding mode control.

6 EXPERIMENTAL RESULTS

Figs. 15 and 16 show the experimental evolution of the position and the experimental phase plane trajectory when the sliding condition is just validated: |c_| = |s1p|. It can be seen that the step reference of 400 steps is reached for t = 0.08 s without overshoot or steady-state error and that there are almost three commutations to reach the reference.

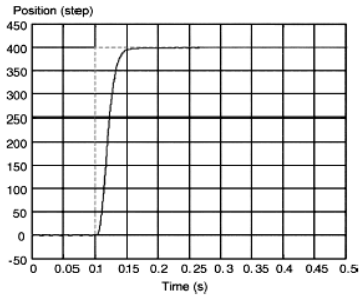


Figure 15: Position and reference (400 steps).

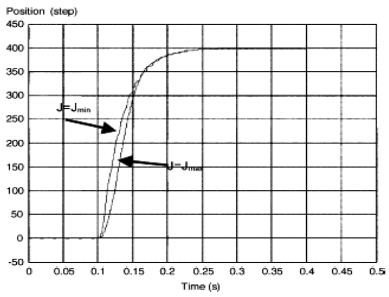


Figure 16: Position and reference for two different inertias.

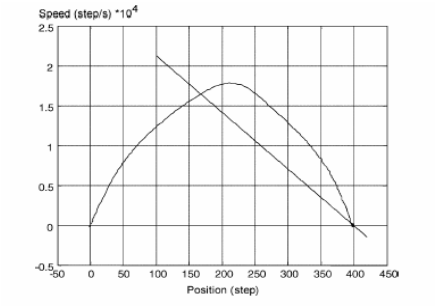


Figure 17: Phase plane trajectory.

Figs. 17 and 18 show the evolution of the position and the phase plane trajectories for two different inertias (J = Jmin = 0.0023 kg.m^2 and J = Jmax = 0.013485 kg.m^2).

By comparing the two responses, it can be noted that the reference is always reached without any overshoot or steady-state error whatever the inertia of the drive.

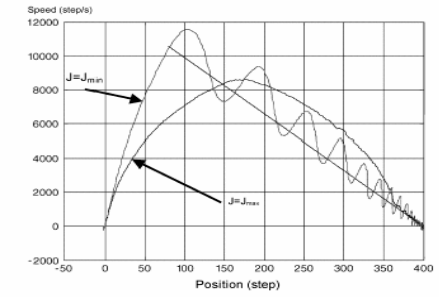


Figure 18: Phase plane trajectory for two different inertias.

From these results, it can be seen that the robustness of the proposed approach between external disturbances and plant parameter variations is experimentally validated.

7 CONCLUSION

In this paper, we have shown that by using a sliding mode control applied to an IFOC, a high-precision positioning of an IM shaft can be achieved whatever the mechanical configuration of the load is. Indeed, the position reference is obtainedwithout any overshoot or static errorwhatever the inertia or the load torque are. Furthermore, it has been shown that the chattering problem around the switching surface can be alleviated using the VSC approach with LFSG. Therefore, the proposed solution can be considered very suitable for induction drive used in robotics or in numerical control of machine tools.

REFERENCES

M.O. Mahmoudi et al, Cascade sliding mode control of a field oriented induction machine drive, EDP Sciences (1999).

Guy Grellet, Guy Clerc, Actionneurs elctrique (EYROLLES, France, 1997).

Vadim I.Utkin, IEEE Transactions on industrial electronics (Vol 40, No 1, 1993)

H. Bühler, Réglage par mode de glissement (Presses polytechniques romandes, Switzerland, 1986).

F. Chen and M. W. Dunnigan, "Sliding-mode torque and flux control of an induction machine," *Proc. IEE Electric Power Appl.*, vol. 150, no. 2, pp. 227–236, Mar. 2003.

P. DeWit, R. Ortega, and I. Mareels, "IFOC of inductionmotors is robustly globally stable," *Automatica*, vol. 32, no. 10, Oct. 1996.

Abdel Faqir, "Position Control of an Induction Machine Using Variable Structure Control", thesis (2003).

S. Ferreira Pinto, Sliding mode control of matrix converters with lead-lag power factor, EPE (2001).

E. Etien, Real time induction motor drive using sliding mode linearization, EPE (2001).

Jesus Arellano-Padilla, Robust fuzzy-sliding mode control for motor drives operating with variable loads and pre-defined system noise limits, EPE (2001).

LIST OF PRINCIPAL SYMBOLS

p : Number of pole pairs.

R_s, R_r : Stator and rotor resistance.

L_s, L_r : Stator and rotor inductance.

T_r : Rotor time constant.

L_m : Magnetizing inductance.

i_{ds}, i_{qs} : Stator currents in d-q rotating reference frame.

v_{ds}, v_{qs} : Stator voltages d-q rotating reference frame.

ϕ_{dr}, ϕ_{qr} : Rotor fluxes d-q rotating reference frame.

ω_r : Rotor speed.

T_e : Electromagnetic torque.

θ_s : Angular position.

s : Laplace operator (d/dt).

σ : Coefficient of dispersion.

J : Total rotor inertia constant.

MULTIVARIATE CONTROL CHARTS WITH A BAYESIAN NETWORK

Sylvain Verron, Teodor Tiplica and Abdessamad Kobi
LASQUO/ISTIA, 62, avenue Notre Dame du Lac, 49000, Angers, France
sylvain.verron@univ-angers.fr, teodor.tiplica@univ-angers.fr, abdessamad.kobi@univ-angers.fr

Keywords: SPC, Bayesian network, multivariate control charts, T^2, MEWMA.

Abstract: The purpose of this article is to present an approach allowing the fault detection of a multivariate process with a bayesian network. As a discriminant analysis is easily modeled with a bayesian network, we will show that we we can consider the multivariate T^2 and MEWMA control charts as particular cases of the discriminant analysis. So, we give the structure of the bayesian network as well as the parameters of the network in order to detect faults in the multivariate space in the same manners as if we used multivariate control charts. The resulting bayesian network, with a computed threshold, is similar to the multivariate control charts.

1 INTRODUCTION

Nowadays, process control (or process monitoring) is becoming an essential task. Indeed, processes are increasingly complex and automatized (containing a lot of sensors and actuators). As a consequence, the control of these processes is more and more difficult. In (Chiang et al., 2001), authors give two principal approaches to perform the process control, namely, data-driven techniques and analytical techniques. The analytical technique is, in theory, the better approach. It is based on analytical (physical) models of the system and permits to simulate the system. Though, at each instant, the theoretical value of each sensor can be known for the normal operating state of the system. As a consequence, it is relatively easy to see if the real process values are similar to the theoretical values. But, the major drawback of this approach is the fact that it requires detailed models of the process. An effective detailed model can be very difficult, time consuming and expensive to obtain, particularly for large-scale systems with many variables. The data-driven approaches are a family of different techniques based on the analysis of the real data extracted from the process. These methods are based on rigorous statistical development of the process data. We can principally cite control charts, methods based on Principal Component Analysis (PCA), Projection to Latent Structure (PLS) or Discriminant Analysis (DA) (Chiang et al., 2001). The process control can be viewed as a three-step procedure: the first one is the fault detection that concludes on the presence of a disturbance in the process; secondly, it is needed to diagnosis the disturbance (fault diagnosis); finally, the process must return in normal operation state (process recovery).

Many data-driven techniques for the fault detection can be found in the literature: univariate statistical process control (Shewhart charts) (Shewhart, 1931), multivariate statistical process control (T^2,Q MEWMA, MCUSUM charts) (Hotelling, 1947; Lowry et al., 1992; Pignatiello and Runger, 1990), and some PCA (Principal Component Analysis) based techniques (Jackson, 1985) like Moving PCA (Bakshi, 1998). In (Kano et al., 2002), authors make comparisons between these different techniques. Other important approaches are PLS (Projection to Latent Structures) based approaches (MacGregor and Kourti, 1995).

These fault detection techniques are able to detect a fault (a disturbance) in a multivariate process. The fault can be a shift of the mean affecting one or more variables. For example, it can be a step or a trend. The major drawback of these techniques is that they do not give any indication about the root cause of a detected fault, and so they are not fully exploitable in

the industry. In order to accomplish the fault diagnosis, many approaches have been proposed (Kourti and MacGregor, 1996). The fault diagnosis procedure can also be seen as a classification task. Indeed today's processes give many measurements that can be stored in a database when the process is in control, but also in case of identified out-of-control states. Many classifiers have been developed. We can cite discriminant analysis like FDA (Fisher Discriminant Analysis) (Duda et al., 2001), SVM (Support Vector Machine) (Vapnik, 1995), kNN (k-nearest neighborhood) (Cover and Hart, 1967), ANN (Artificial Neural Networks) (Duda et al., 2001) and bayesian classifiers (Friedman et al., 1997). The performances of these classifiers are reduced in the space described by all the variable of the process. So, before the classification, a feature selection is often required in order to obtain better performances.

We can see that the methods of detection and diagnosis are numerous. Moreover, all these methods are heterogeneous. But, as their final goal is the same (process recovery), they are complementary. So, it will be of interest to have the possibility to detect and to diagnose a fault with a single tool. An interesting approach for the diagnosis can be the use of Bayesian Networks (BN) (Friedman et al., 1997). In this article, we will study a possibility to detect a fault in a multivariate process by modelling the multivariate control chart with a BN. So, detection and diagnosis of a fault would be possible on a same tool: a bayesian network.

The article is structured as follows: in the second section, we will present the utilization of the multivariate control charts for the detection of faults in a multivariate process; the third section presents some aspects on bayesian networks and particularly on bayesian network classifiers; the fourth section gives the procedure to model some multivariate control charts (T^2 and MEWMA), with a bayesian network. In the last section, we conclude on the proposed approach and give some outlooks.

2 MULTIVARIATE CONTROL CHARTS

2.1 The Hotelling T^2 Control Chart

The first work in the field of fault detection in multivariate processes began in 1947 with Hotelling (Hotelling, 1947). He was the first to propose a multivariate control chart based on a statistical distance. For a process with p variables, we can write the statistic T^2 as:

$$T^2 = (x-\mu)^T \Sigma^{-1} (x-\mu) \quad (1)$$

where: x is the observation vector of size $1 \times p$, μ is the mean vector of size $1 \times p$, Σ is the variance-covariance matrix of size $p \times p$ and the symbol ^{T} represents the transpose of a vector or a matrix.

As we can see in the equation 1, the statistic T^2 is a scalar. So, we can plot the value of the T^2 for different time instants, and with an appropriate control limit (computed by taking into account statistical considerations), the T^2 control chart is obtained. On this chart, each point represents the information extracted form all the p variables. A fault is detected when a point is beyond the control limit.

As for the univariate Shewhart control chart, the set up of the T^2 chart is made in two phases. During the first phase, parameters of the process (μ and Σ) are estimated. Concerning the computation of these parameters estimations and the computation of the control limit, readers can refer to the book of Montgomery (Montgomery, 1997). Once the parameters are estimated, the T^2 control chart can be drawn. It is very important to verify that the process is in control during the first phase. The second phase represents the real monitoring of the process on the assumption of a multivariate normal distribution.

2.2 The Mewma Control Chart

As in the case of the univariate Shewhart control chart, the major drawback of the T^2 control chart is his moderate performances to detect small mean shifts. In order to solve this problem, other multivariate control charts have been proposed: MEWMA (Multivariate Exponentially Weighted Moving Average) (Lowry et al., 1992) and MCUSUM (Multivariate CUmulative SUM) (Pignatiello and Runger, 1990). These charts are respectively the multivariate analogous of the EWMA and CUSUM control charts. The principle of the MEWMA control chart is to take into account the process evolution in weighting past observations extracted from the process, as indicated in the equation 2:

$$y_t = \lambda x_t + (I-\lambda) y_{t-1} \quad (2)$$

where λ is a $p \times p$ diagonal weighting matrix, I is the $p \times p$ identity matrix, x_t is the observation vector (size $1 \times p$) at instant t, $y_0 = \mu$ is the mean vector (size $1 \times p$) of the p variables.

Based on the same principle than a T^2 control chart, one can monitor the process with the statistic given in the equation 3:

$$T_t^2 = y_t^T \Sigma_y^{-1} y_t \quad (3)$$

where y_t is the transformed observation vector at instant t, Σ_y^{-1} is the inverse of the variance-covariance matrix of y_t. It can be concluded that the process is out of control as soon as the T_t^2 crosses the control limit. Bodden (Bodden and Rigdon, 1999) proposed an algorithm to find the control limit in order to respect a given number of false alarm and a given λ.

These multivariate control charts (T^2 and MEWMA) are efficient to detect a fault in a multivariate process. But, as we have already said, the fault diagnosis cannot be made in the same time. So, it seems to be an interesting approach to do the fault detection and the fault diagnosis of a multivariate process by using a single tool. Several works demonstrated that bayesian networks are able to diagnose correctly the fault of a multivariate process. So, it is of interest to study if a bayesian network can accomplish the fault detection in an efficient way.

3 BAYESIAN NETWORK CLASSIFIERS

A Bayesian Network (BN) (Jensen, 1996; Pearl, 1988) is an acyclic graph where each variable is a node (that can be continuous or discrete). Edges of the graph represent dependence between linked nodes. A formal definition is given here:

A bayesian network is a triplet $\{\mathbf{G, E, D}\}$ where:

$\{\mathbf{G}\}$ is a directed acyclic graph, $\mathbf{G} = (V,A)$, with V the ensemble of nodes of $\mathbf{G}$, and A the ensemble of edges of $\mathbf{G}$,

$\{\mathbf{E}\}$ is a finite probabilistic space (Ω, Z, p), with Ω a non-empty space, Z a collection of subspace of Ω, and p a probability measure on Z with $p(\Omega) = 1$,

$\{\mathbf{D}\}$ is an ensemble of random variables associated to the nodes of $\mathbf{G}$ and defined on $\mathbf{E}$ such as:

$$p(V_1, V_2, \ldots, V_n) = \prod_{i=1}^{n} p(V_i | C(V_i)) \qquad (4)$$

with $C(V_i)$ the ensemble of causes (parents) of V_i in the graph $\mathbf{G}$.

Bayesian network classifiers are particular BN (Friedman et al., 1997). They always have a discrete node C coding the k different classes of the system. Thus, other variables $X_1, \ldots, X_p$ represent the p descriptors (variables) of the system.

A famous bayesian classifier is the Naïve Bayesian Network (NBN), also named Bayes classifier (Langley et al., 1992). This bayesian classifier makes the strong assumption that the descriptors of the system are class conditionally independent. Assuming the hypothesis of normality of each descriptor, the NBN is equivalent to the classification rule of the diagonal quadratic discriminant analysis. But, in practice, this assumption of independence and non correlated variables is not realistic. In order to deal with correlated variables, several approaches have been developed. We can cite the Tree Augmented Naïve bayesian networks (TAN) (Friedman et al., 1997). These BNs are based on a NBN but a tree is added between the descriptors. An other interesting approach is the Kononenko one (Kononenko, 1991), which represent some variables in one node. As in (Perez et al., 2006) the assumption we will make is that this variable follows a normal multivariate distribution (conditionally to the class) and we will refer to this kind of BN as Condensed Semi Naïve Bayesian Network (CSNBN).

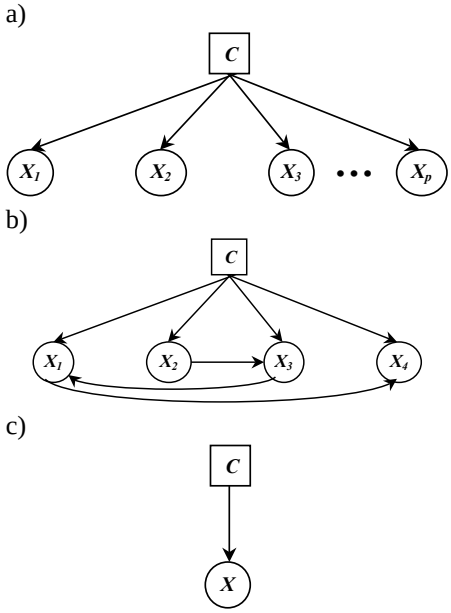


Figure 1: Different bayesian network classifiers: NBN (a), TAN (b) and CSNBN (c).

4 MULTIVARIATE CONTROL CHARTS MODELIZED WITH BAYESIAN NETWORK

The purpose of this article is to present a method allowing the fault detection in a multivariate process with the use of a BN. In the section 2, we have presented the multivariate control charts T^2 and MEWMA. It can be remarked that for these two charts, one needs to compute a statistical distance on

the form of $X^T\Sigma^{-1}X$ which is known as the square Mahalanobis distance. This distance is also used in a Discriminant Analysis (DA).

Indeed, one can distinguish two aspects of discriminant analysis: the predictive and the classification aspect. Given a system with p variables (descriptors) and with k identified classes, the predictive discriminant analysis (or Fisher discriminant analysis) is generally used to find $k-1$ new descriptors of a system. These new $k-1$ descriptors (which are a linear combination of the original descriptors) are supposed to maximally discriminate between the k identified classes of the system. The other aspect of the discriminant analysis is the classification one. The purpose of the classification is principally to allocate a new observation to one of the k identified classes of the system. In the remainder of the article, the Discriminant Analysis (DA) will be viewed only as a supervised classification method (Duda et al., 2001).

DA is based on the Bayes decision rule. This rule allocates a new observation x to the class C_i with the maximum a posteriori probability $p(C_i|x)$ giving the value of each descriptor, as defined in the equation 5.

$$x \in C_i,\ if\ i = \underset{i=1,\ldots,k}{\operatorname{argmax}}\{p(C_i|x)\} \tag{5}$$

This decision rule is named "Bayes decision rule" because it is based on the Bayes rule which gives the value of $p(C_i|x)$ as stated in 6.

$$P(C_i|x) = \frac{p(C_i)p(x|C_i)}{p(x)} \tag{6}$$

where $p(C_i)$ is the marginal probability of belonging to the class C_i. We can see that for each class, the denominator of the equation 6 is the same, so, it is not implicated in the discriminant function. Then, equation 5 can be rewritten as:

$$x \in C_i,\ if\ i = \underset{i=1,\ldots,k}{\operatorname{argmax}}\{p(C_i)p(x|C_i)\} \tag{7}$$

In industrial systems, it is usual to assume that data follow multivariate normal distribution. The density function conditionally to a class C_i can be written as in equation 8, where μ_i is the mean vector of the class C_i, Σ_i is the covariance matrix of the class C_i and $|\Sigma_i|$ represents the determinant of the matrix Σ_i.

$$\phi(x|C_i) = \frac{\exp(-\frac{1}{2}(x-\mu_i)^t\Sigma_i^{-1}(x-\mu_i))}{(2\pi)^{p/2}|\Sigma_i|^{1/2}} \tag{8}$$

We remind that, for n samples, the Maximum Likelihood Estimation (MLE) gives (Duda et al., 2001):

$$\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{9}$$

and:

$$\hat{\Sigma} = \frac{1}{n-1}\sum_{i=1}^{n}(x_i-\hat{\mu})(x_i-\hat{\mu})^t \tag{10}$$

So, if our system has k classes, the different probabilities can be computed with equation 11.

$$p(x|C_i) = \frac{\phi(x|C_i)}{\sum_{j=1}^{k} p(C_j)\phi(x|C_j)} \tag{11}$$

So, the equation 7 can be rewritten by the equation 12.

$$x \in C_i,\ if\ i = \underset{i=1,\ldots,k}{\operatorname{argmax}} \left\{ \frac{p(C_i)\phi(x|C_i)}{\sum_{j=1}^{k}\phi(x|C_j)} \right\} \tag{12}$$

At the first view, we can tell that the DA is well suited for the fault diagnosis of industrial systems. But, we will show that we can also realize the fault detection methods of the multivariate control charts with a DA, and particularly with a BN. So, to do that we will consider the multivariate control charts as a DA where the goal is to attribute a class (in or out of control) to a new observation of the system. Realising a DA with a BN is very simple. The structure of the network will be composed of a class node (discrete variable) with two modalities (in or out of control). In order to represent the different descriptors of the system, we use only one normal multivariate variable. This normal multivariate variable will represent the x for the T^2 chart and the y_t for the MEWMA chart. To resume, we will obtain a structure similar to the CSNBN.

However, an essential question we remain concerning the parameters of the network. We have to fix a normal multivariate distribution for each classes of the problem. The distribution of the "in control" state will be $N(\mu,\Sigma)$. In order to obtain a decision boundary between the two states of the process, we fix the distribution of the "out of control" state to $N(\mu, c\times\Sigma)$, where c is a coefficient different than 1. These two distributions will have same center (μ) and same shape (because $c\times\Sigma$ is simply a scale extending of the shape of Σ). The difference between the two states can be expressed as: scattering about the "out of control" state (OC) is more important than scattering about the "in control" state (IC). We can represent

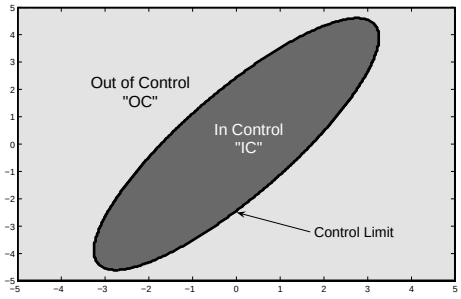


Figure 2: Example of the classification areas.

the classification areas of the two states of the system for a bivariate example like in the figure 2.

Usually, in the frame of Statistical Process Control (SPC) techniques, one has to choose a risk factor α in order to correctly control the process. In the case of the BN, we have to fix a probability threshold τ allowing to not badly reject some situations "in control" (false alarms).

Otherwise the threshold τ is computed in order to have similar classification areas than those of the multivariate control chart. We will name *CL* the control limit for the control charts and we supposed that *CL* is known for each chart, see (Montgomery, 1997; Lowry et al., 1992). The τ probability limit corresponds to the probability, for the class "OC", of the x_{CL} value which is an observation giving a T^2 equal at the control limit *CL*. We will assume that the marginal probabilities of the two classes are unknown and so are fixed to 0.5 each. So, $p(IC) = p(OC) = 0.5$.

$$\begin{aligned}\tau &= p(IC|x_{CL}) \\ &= \frac{p(OC)\phi(x_{CL}|OC)}{p(IC)\phi(x_{CL}|IC)+p(OC)\phi(x_{CL}|OC)} \\ &= \frac{\phi(x_{CL}|OC)}{\phi(x_{CL}|IC)+\phi(x_{CL}|OC)} \end{aligned} \quad (13)$$

By using equation 8 in the previous equation, and after simplification, we will obtain the equation of τ as:

$$\tau = \frac{\frac{\exp(-0.5\frac{CL}{c})}{c^{p/2}}}{\exp(-0.5CL)+\frac{\exp(-0.5\frac{CL}{c})}{c^{p/2}}} \quad (14)$$

In conclusion, for the BN, we will obtain the following decision rule: if $p(OC|x) \geq \tau$, then the process is out-of-control (OC), otherwise the process is in control (IC).

In order to use the previous result, we have taken a simple example of a 5 variables system. The parameters of the system is given as:

$$\begin{aligned}\mu &= [5\,10] \\ \Sigma &= \begin{pmatrix} 1 & 1.2 \\ 1.2 & 2 \end{pmatrix}\end{aligned}$$

We have simulated the system during 30 observations, but at the observation number 6, a step of magnitude 0.5 is introduced in the process. In the figure 3 we are presenting the detection of the fault with the control chart (upper graph) and with the BN approach (lower graph). In a similar way, the figure 4 represents the result of the MEWMA approach.

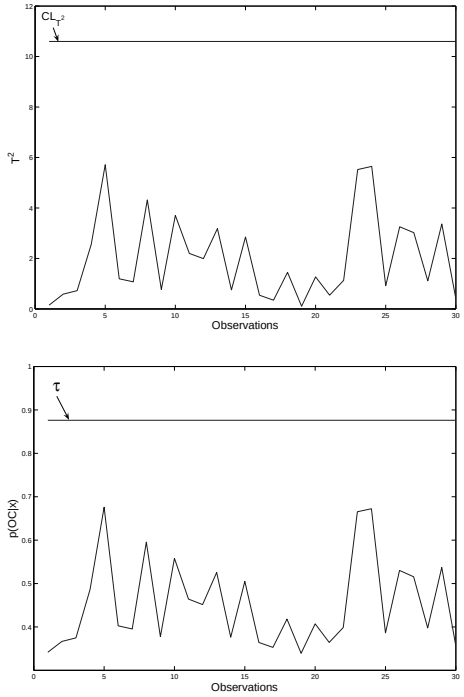


Figure 3: Results with the T^2 chart and the T^2 chart by BN apporach.

On the figure 3 and 4, we can see that for each multivariate charts, the BN approach gives the same results than the specified chart (with a scaling factor). So, we can conclude that the different multivariate control charts (T^2 and MEWMA) can be easily modeled with a BN.

5 CONCLUSIONS AND OUTLOOKS

In this article, after an introduction on the process control (and particularly on the data-driven techniques for the detection and the diagnosis), we have

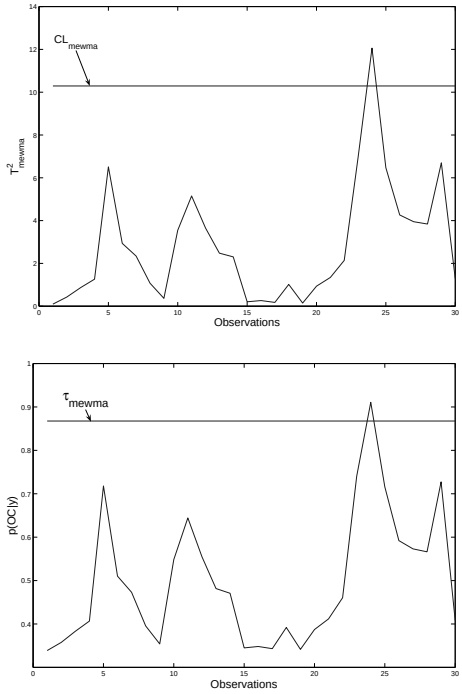


Figure 4: Results with the MEWMA chart and the MEWMA chart by BN approach.

outlined the problem that there is not a single tool able to perform both: the fault detection and the fault diagnosis of a system. As a bayesian network can be an efficient way to diagnosis a fault, it is natural to see how this tool can also execute the fault detection task. We have selected two fault detection techniques: the T^2 chart and the MEWMA chart. We have demonstrated that these charts can be viewed as a discriminant analysis and so can be implemented in a simple bayesian network.

The evident outlook to this work is the full study of the use of BN in order to monitor and control a multivariate process.

REFERENCES

Bakshi, B. R. (1998). Multiscale PCA with application to multivariate statistical process monitoring. *AIChE Journal*, 44(7):1596–1610.

Bodden, K. and Rigdon, S. (1999). A program for approximating the in-control arl for the mewma chart. *Journal of Quality Technology*, 31(1):120–123.

Chiang, L. H., Russell, E. L., and Braatz, R. D. (2001). *Fault detection and diagnosis in industrial systems*. New York: Springer-Verlag.

Cover, T. and Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13:21–27.

Duda, R. O., Hart, P. E., and Stork, D. G. (2001). *Pattern Classification 2nd edition*. Wiley.

Friedman, N., Geiger, D., and Goldszmidt, M. (1997). Bayesian network classifiers. *Machine Learning*, 29(2-3):131–163.

Hotelling, H. (1947). Multivariate quality control. *Techniques of Statistical Analysis*, :111–184.

Jackson, E. J. (1985). Multivariate quality control. *Communication Statistics - Theory and Methods*, 14:2657 – 2688.

Jensen, F. V. (1996). *An introduction to Bayesian Networks*. Taylor and Francis, London, United Kingdom.

Kano, M., Nagao, K., Hasebe, S., Hashimoto, I., Ohno, H., Strauss, R., and Bakshi, B. (2002). Comparison of multivariate statistical process monitoring methods with applications to the eastman challenge problem. *Computers and Chemical Engineering*, 26(2):161–174.

Kononenko, I. (1991). Semi-naive bayesian classifier. In *EWSL-91: Proceedings of the European working session on learning on Machine learning*, pages 206–219.

Kourti, T. and MacGregor, J. F. (1996). Multivariate spc methods for process and product monitoring. *Journal of Quality Technology*, 28(4):409–428.

Langley, P., Iba, W., and Thompson, K. (1992). An analysis of bayesian classifiers. In *National Conference on Artificial Intelligence*.

Lowry, C. A., Woodall, W. H., Champ, C. W., and Rigdon, S. E. (1992). A multivariate exponentially weighted moving average control chart. *Technometrics*, 34(1):46–53.

MacGregor, J. and Kourti, T. (1995). Statistical process control of multivariate processes. *Control Engineering Practice*, 3(3):403–414.

Montgomery, D. C. (1997). *Introduction to Statistical Quality Control, Third Edition*. John Wiley and Sons.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers.

Perez, A., Larranaga, P., and Inza, I. (2006). Supervised classification with conditional gaussian networks: Increasing the structure complexity from naive bayes. *International Journal of Approximate Reasoning*, 43:1–25.

Pignatiello, J. and Runger, G. (1990). Comparisons of multivariate cusum charts. *Journal of Quality Technology*, 22(3):173–186.

Shewhart, W. A. (1931). *Economic control of quality of manufactured product*. New York : D. Van Nostrand Co.

Vapnik, V. N. (1995). *The Nature of Statistical Learning Theory*. Springer.

AN INTELLIGENT MARSHALING PLAN BASED ON MULTI-POSITIONAL DESIRED LAYOUT IN CONTAINER YARD TERMINALS

Yoichi Hirashima
Dept. Media Science, Fac. Information Science and Technology, Osaka Institute of Technology
1-79-1 Kitayama, Hirakata-City, Osaka, 573-0196 Japan
hirash-y@is.oit.ac.jp

Keywords: Container marshaling, Block stacking problem, Q-learning, Reinforcement learning, Binary tree.

Abstract: This paper proposes a new scheduling method for a marshaling in the container yard terminal. The proposed method is derived based on Q-Learning algorithm considering the *desired position* of containers that are to be loaded into a ship. In the method, 3 processes can be optimized simultaneously: rearrangement order of containers, layout of containers assuring explicit transfer of container to the *desired position*, and removal plan for preparing the rearrange operation. Moreover, the proposed method generates several desired positions for each container, so that the learning performance of the method can be improved as compared to the conventional methods. In general, at container yard terminals, containers are stacked in the arrival order. Containers have to be loaded into the ship in a certain order, since each container has its own shipping destination and it cannot be rearranged after loading. Therefore, containers have to be rearranged from the initial arrangement into the desired arrangement before shipping. In the problem, the number of container-arrangements increases by the exponential rate with increase of total count of containers, and the rearrangement process occupies large part of total run time of material handling operation at the terminal. For this problem, conventional methods require enormous time and cost to derive an admissible result. In order to show effectiveness of the proposed method, computer simulations for several examples are conducted.

1 INTRODUCTION

In recent years, the number of shipping containers grows rapidly, and operations for layout-rearrangement of container stacks occupy a large part of the total run time of shipping at container terminals. Since containers are moved by a transfer crane driven by human operator, and thus, the container operation is important to reduce cost, run time, and environmental burden of material handling systems (Siberholz et al., 1991). Commonly, materials are packed into containers and each container has its own shipping destination. Containers have to be loaded into a ship in a certain desired order because they cannot be rearranged in the ship. Thus, containers must be rearranged before loading if the initial layout is different from the desired layout. Containers carried in the terminal are stacked randomly in a certain area called bay and a set of bays are called yard. When the number of containers for shipping is large, the rearrangement operation is complex and takes long time to achieve the desired layout of containers. Therefore the rearrangement process occupies a large part of the total run time of shipping. The rearrangement process conducted within a bay is called marshaling.

In the problem, the number of stacks in each bay is predetermined and the maximum number of containers in a stack is limited. Containers are moved by a transfer crane and the destination stack for the container in a bay is selected from the stacks being in the same bay. In this case, a long series of movements of containers is often required to achieve a desired layout, and results (the number of container-movements) that are derived from similar layouts can be quite different. Problems of this type have been solved by using techniques of optimization, such as genetic algorithm (GA) and multi agent method (Koza, 1992; Minagawa and Kakazu, 1997). These methods can successfuly yield some solutions for block stacking problems. However, they adopt the environmental model different from the marshaling process, and do not assure to obtain the desired layout of containers.

The Q-learning (Watkins and Dayan, 1992) is known to be effective for learning under unknown environment. In the Q-learning for generating marshaling plan, all the estimates of evaluation-values for pairs of the layout and movement of containers are calculated. These values are called “Q-value” and Q-

table is a look-up table that stores Q-values. The input of the Q-table is the plant state and the output is a Q-value corresponding to the input. A movement is selected with a certain probability that is calculated by using the magnitude of Q-values. Then, the Q-value corresponding to the selected movement is updated based on the result of the movement. The optimal pattern of container movements can be obtained by selecting the movement that has the largest Q-value at each state-movement pair, when Q-values reflect the number of container movements to achieve the desired layout. However, conventional Q-table has to store evaluation-values for all the state-movement pairs. Therefore, the conventional reinforcement learning method, Q-learning, has great difficulties for solving the marshaling problem, due to its huge number of learning iterations and states required to obtain admissible operation of containers (Baum, 1999). Recently, a Q-learning method that can generate marshaling plan has been proposed (Hirashima et al., 1999). Although these methods were effective several cases, the desired layout was not achievable for every trial so that the early-phase performances of learning process can be degraded.

In this paper, a new reinforcement learning system to generate a marshaling plan is proposed. The learning process in the proposed method is consisted of two stages: ① determination of rearrangement order, ② selection of destination for removal containers. Learning algorithms in these stages are independent to each other and Q-values in one stage are referred from the other stage. That is, Q-values are discounted according to the number of container movement and Q-table for rearrangement is constructed by using Q-values for movements of container, so that Q-values reflect the total number of container movements required to obtain a desired layout. Moreover, in the end of stage ①, selected container is rearranged into the desired position so that every trial can achieve the desired layout. In addition, in the proposed method, each container has several desired positions in the final layout, and the feature is considered in the learning algorithm. Thus, the early-phase performances of the learning process can be improved. Finally, effectiveness of the proposed method is shown by computer simulations for several cases.

2 PROBLEM DESCRIPTION

Fig.1 shows an example of container yard terminal. The terminal consists of containers, yard areas, yard transfer cranes, auto-guided vehicles, and port crane. Containers are carried by trucks and each container is

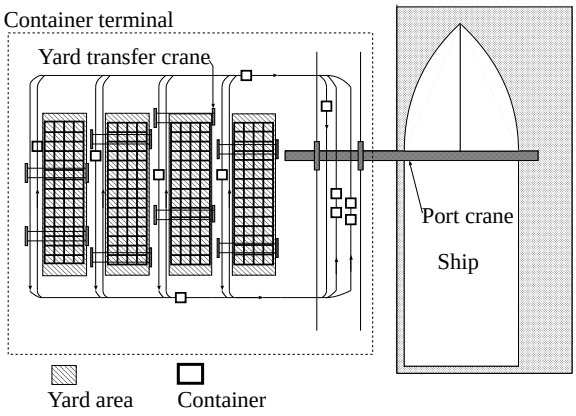


Figure 1: Container terminal.

stacked in a corresponding area called bay and a set of bays constitutes a yard area. Each bay has n_{y} stacks that m_{y} containers can be laden, the number of containers in a bay is k, and the number of bays depends on the number of containers. Each container is recognized by an unique name $\mathrm{c}_i\,(i = 1, \cdots, k)$. A position of each container is discriminated by using discrete position numbers, $1, \cdots, n_{\mathrm{y}} \cdot m_{\mathrm{y}}$. Then, the position of the container c_i is described by $x_i\,(1 \leq i \leq k, 1 \leq x_i \leq m_{\mathrm{y}} \cdot n_{\mathrm{y}})$, and the state of a bay is determined by the vector, $x = [x_1, \cdots, x_k]$.

2.1 Grouping

The desired layout in a bay is generated based on the loading order of containers that are moved from the bay to a ship. In this case, the container to be loaded into the ship can be anywhere in the bay if it is on top of a stack. This feature yields several desired layouts for the bay. In the addressed problem, when containers on different stacks are placed at the same height in the bay, it is assumed that the positions of such containers can be exchanged. Fig.2 shows an example of desired layouts, where $m_{\mathrm{y}} = n_{\mathrm{y}} = 3, k = 9$. In the figure, containers are loaded in the ship in the descendent order. Then, containers $\mathrm{c}_7, \mathrm{c}_8, \mathrm{c}_9$ are in the same group (Group1), and their positions are exchanged because the loading order can be kept unchanged after the exchange of positions. In the same way, $\mathrm{c}_4, \mathrm{c}_5, \mathrm{c}_6$ are in the Group2, and $\mathrm{c}_1, \mathrm{c}_2, \mathrm{c}_3$ are in the Group3 where positions of containers can be exchanged. Consequently several candidates for desired layout of the bay are generated from the original desired-layout.

In addition to the grouping explained above, a "heap shaped group" for n_{y} containers at the top of stacks in original the desired-layout (group 1) is generated as follows:

1. n_{y} containers in group 1 can be placed at any

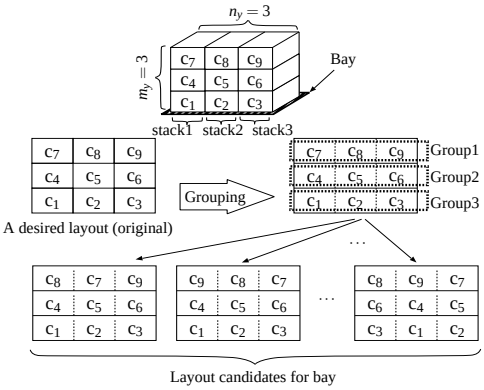


Figure 2: Layouts for bay.

stacks if their height is same as the original one.

2. Each of them can be stacked on other $n_{\mathrm{y}}-1$ containers when both of followings are satisfied:
   (a) They are placed at the top of each stack in the original disired-layout,
   (b) The container to be stacked is loaded into the ship before other containers being under the container.

Other groups are the same as ones in the original grouping, so that the grouping with heap contains all the desired layout in the original grouping.

2.2 Marshaling Process

The marshaling process consists of 2 stages: ① selection of a container to be rearranged, and ② removal of the containers on the selected container in ①. After these stages, rearrangement of the selected container is conducted. In the stage ②, the removed container is placed on the destination stack selected from stacks being in the same bay. When a container is rearranged, n_{y} positions that are at the same height in a bay can be candidates for the destination. In addition, n_{y} containers can be placed for each candidate of the destination. Then, defining t as the time step, $c_{\mathrm{a}}(t)$ denotes the container to be rearranged at t in the stage ①. $c_{\mathrm{a}}(t)$ is selected from candidates $\mathrm{c}_{y_{i_1}}$ $(i_1 = 1,\cdots,n_{\mathrm{y}}^2)$ that are at the same height in a desired layout. A candidate of destination exists at a bottom position that has undesired container in each corresponding stack. The maximum number of such stacks is n_{y}, and they can have n_{y} containers as candidates, since the proposed method considers groups in the desired position. The number of candidates of $c_{\mathrm{a}}(t)$ is thus $n_{\mathrm{y}} \times n_{\mathrm{y}}$. In the stage ②, the container to be removed at t is $c_{\mathrm{b}}(t)$ and is selected from two containers $\mathrm{c}_{y_{i_2}}$ $(i_2 = 1,2)$ on the top of stacks. c_{y_1} is on the $c_{\mathrm{a}}(t)$ and c_{y_2} is on the destination of $c_{\mathrm{a}}(t)$. Then, in the stage ②, $c_{\mathrm{b}}(t)$ is removed to one of the other stacks in the same bay, and the destination stack $u(t)$ at time t is selected from the candidates u_j $(j = 1,\cdots,n_{\mathrm{y}}-2)$. $c_{\mathrm{a}}(t)$ is rearranged to its desired position after all the $\mathrm{c}_{y_{i_2}}$s are removed. Thus, a state transition of the bay is described as follows:

$$x_{t+1} = \begin{cases} f(x_t, c_{\mathrm{a}}(t)) & (\text{stage } ①) \\ f(x_t, c_{\mathrm{b}}(t), u(t)) & (\text{stage } ②) \end{cases} \quad (1)$$

where $f(\cdot)$ denotes that removal is processed and x_{t+1} is the state determined only by $c_{\mathrm{a}}(t), c_{\mathrm{b}}(t)$ and $u(t)$ at the previous state x_t. Therefore, the marshaling plan can be treated as the Markov Decision Process.

Additional assumptions are listed below:

1. The bay is 2-dimensional.
2. Each container has the same size.
3. The goal position of the target container must be located where all containers under the target container are placed at their own goal positions.
4. $k \leq m_{\mathrm{y}} n_{\mathrm{y}} - 2m_{\mathrm{y}} + 1$

The maximum number of containers that must removed before rearrangement of $c_{\mathrm{a}}(t)$ is $2m_{\mathrm{y}}-1$ because the height of each stack is limited to m_{y}. Thus, assumption (4) assures the existence of space for removing all the $c_{\mathrm{b}}(t)$, and $c_{\mathrm{a}}(t)$ can be placed at the desired position from any state x_t.

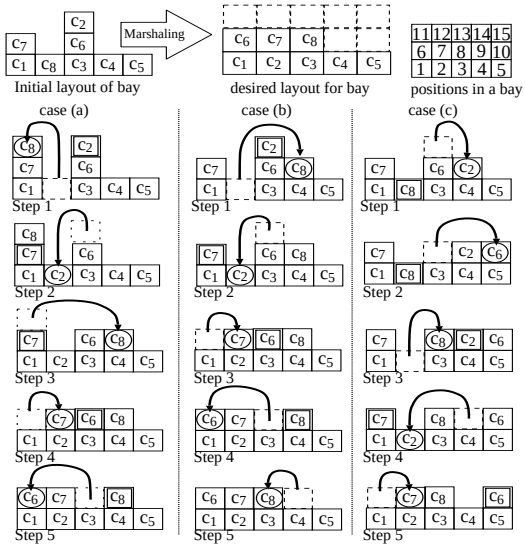


Figure 3: Marshaling process.

Figure 3 shows 3 examples of marshaling process, where $m_{\mathrm{y}} = 3, n_{\mathrm{y}} = 5, k = 8$. Positions of containers are discriminated by integers $1,\cdots,15$. The first container to be loaded is c_8 and containers must be loaded by descendent order until c_1 is loaded. In the figure, a container marked with a □ denotes c_1, a container marked with a ◯ is removed one, and an arrowed

line links source and destination positions of removed container. Cases (a),(b) have the same order of rearrangement, c_2, c_7, c_6, and the removal destinations are different. Whereas, case (c) has the different order of rearrangement, c_8, c_2, c_7. When no groups are considered in desired arrangement, case (b) requires 5 steps to complete the marshaling process, and other cases require one more step. Thus, the total number of movements of container can be changed by the destination of the container to be removed as well as the rearrangement order of containers.

If groups are considered in desired arrangement, case (b) achieves a goal layout at step2, case (a) achieves at step3, case (c) achives at step4. If extended groups are considered, cases (a),(b) achive goal layouts at step2 and case (c) achives at step4. Since extended goal layouts include the non-extended goal layouts, and since non-extended goal layouts include a non-grouping goal layout, equivalent or better marshaling plan can be generated by using the extended goal notion as compared to plans generated by other goal notions.

The objective of the problem is to find the best series of movements which transfers every container from an initial position to the goal position. The goal state is generated from the shipping order that is predetermined according to destinations of containers. A series of movements that leads a initial state into the goal state is defined as an episode. The best episode is the series of movements having the smallest number of movements of containers to achieve the goal state.

3 REINFORCEMENT LEARNING FOR MARSHALING PLAN

3.1 Update Rule of Q-values

In the selection of c_a, the container to be rearranged, an evaluation value is used for each candidate $c_{y_{i_1}}$ $(i_1 = 1, \cdots, n_y^2)$. In the same way, evaluation values are used in the selection of the container to be removed c_b and its destination u_j $(j = 1, \cdots, n_y - 2)$. Candidates of c_b is $c_{y_{i_2}}$ $(i_2 = 1, \cdots, n_y)$. The evaluation value for the selection of $c_{y_{i_1}}$, $c_{y_{i_2}}$ and u_j at the state x are called Q-values, and a set of Q-values is called Q-table. At the lth episode, the Q-value for selecting $c_{y_{i_1}}$ is defined as $Q_1(l, x, c_{y_{i_1}})$, the Q-value for selecting $c_{y_{i_2}}$ is defined as $Q_2(l, x, c_{y_{i_1}}, c_{y_{i_2}})$ and the Q-value for selecting u_j is defined as $Q_3(l, x, c_{y_{i_1}}, c_{y_{i_2}}, u_j)$. The initial value for both Q_1, Q_2, Q_3 is assumed to be 0.

In this method, a large amount of memory space is required to store all the Q-values referred in every episode. In order to reduce the required memory size, the length of episode that corresponding Q-values are stored should be limited, since long episode often includes ineffective movements of container. In the following, update rule of Q_3 is described. When a series of n movements of container achieves the goal state x_n from an initial state x_0, all the referred Q-values from x_0 to x_n are updated. Then, defining L as the total counts of container-movements for the corresponding episode, L_{min} as the smallest value of L found in the past episodes, and s as the parameter determining the threshold, Q_3 is updated when $L < L_{min} + s\,(s > 0)$ is satisfied by the following equation:

$$\begin{aligned} &Q_3(l, x_t, c_a(t), c_b(t), u(t)) = \\ &\quad (1-\alpha) Q_3(l-1, x_t, c_a(t), c_b(t), u(t)) \\ &\quad +\alpha[R + V_{t+1}] \\ &V_t = \begin{cases} \gamma \max_{y_{i_1}} Q_1(l, x_t, c_{y_{i_1}}) & (\text{stage } ①) \\ \gamma \max_{y_{i_2}} Q_2(l, x_t, c_a(t), c_{y_{i_2}}) & (\text{stage } ②) \end{cases} \end{aligned} \tag{2}$$

where γ denotes the discount factor and α is the learning rate. Reward R is given only when the desired layout has been achieved. L_{min} is assumed to be infinity at the initial state, and updated when $L < L_{min}$ by the following equation: $L = L_{min}$.

In the selection of $c_b(t)$, the evaluation value $Q_3(l, x,\ c_a(t), c_b(t), u_j)$ can be referred for all the u_j $(j = 1 \cdots n_y - 2)$, and the state x does not change. Thus, the maximum value of $Q_3(l, x,\ c_a(t), c_b(t), u_j)$ is copied to $Q_1(l, x, c(t))$, that is,

$$\begin{aligned} &Q_2(l, x, c_a(t), c_b(t)) = \\ &\quad \max_j Q_3(l, x, c_a(t), c_b(t), u_j). \end{aligned} \tag{3}$$

In the selection of $c_a(t)$, the evaluation value $Q_1(l, x,\ c_a(t))$ is updated by the following equations:

$$\begin{aligned} &Q_1(l, x_t, c_a(t)) = \\ &\begin{cases} \max_{y_{i_1}} Q_1(l, x_t, c_{y_{i_1}}) + R & (\text{stage } ①) \\ \max_{y_{i_2}} Q_2(l, x_t, c_a(t), c_{y_{i_2}}) & (\text{stage } ②) \end{cases} \end{aligned} \tag{4}$$

In order to select actions, the "ε-greedy" method is used. In the "ε-greedy" method, $c_a(t), c_b(t)$ and a movement that have the largest $Q_1(l, x, c_a(t))$, $Q_2(l,\ x, c_a(t), c_b(t))$ and $Q_3(l, x, c_a(t), c_b(t), u_j)$ are selected with probability $1-\varepsilon (0 < \varepsilon < 1)$, and with probability ε, a container and a movement are selected randomly.

3.2 Learning Algorithm

By using the update rule, restricted movements and goal states explained above, the learning process is described as follows:

[1]. Count the number of containers being in the goal positions and store it as n

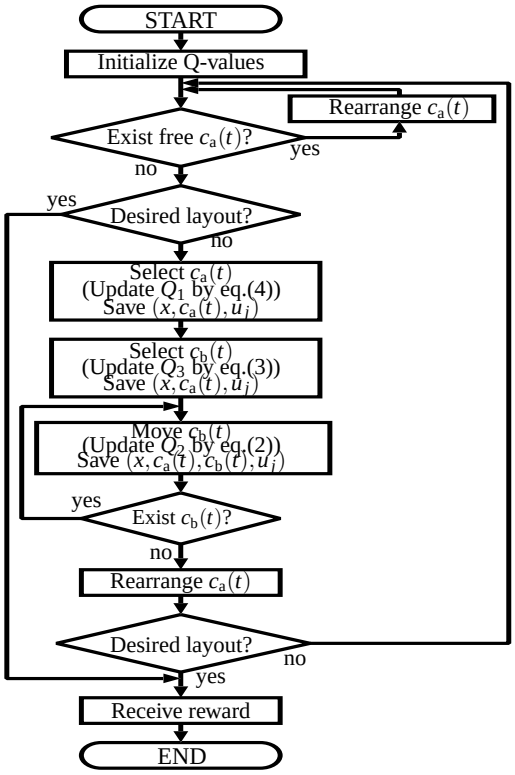


Figure 4: Flowchart of the learning algorithm.

[2]. If $n = k$, go to [10]

[3]. Select $c_a(t)$ to be rearranged

[4]. Store $(x, c_a(t))$

[5]. Select $c_b(t)$ to be removed

[6]. Store $(x, c_a(t), c_b(t))$

[7]. Select destination position u_j for $c_b(t)$

[8]. Store $(x, c_a(t), c_b(t), u_j)$

[9]. Remove $c_b(t)$ and go to [5] if another $c_b(t)$ exists, otherwise go to [1]

[10]. Update all the Q-values referred from the initial state to the goal state according to eqs. (2), (3)

A flow chart of the learning algorithm is depicted in Figure 4.

4 SIMULATIONS

Computer simulations are conducted for 2 cases, and learning performances are compared for following two methods:

(A) proposed method considering grouping with heap,

(B) proposed method considering original grouping,

(C) a learning method using eqs. (2)-(4) as the update rule without grouping (Hirashima et al., 2005),

(D) method (E) considering original grouping.

(E) a learning method using, eqs. (2),(3) as the update rule, which has no selection of the desired position of $c_a(t)$ (Motoyama et al., 2001).

In methods (D),(E), although the stage ② has the same process as in the method (A), the container to be rearranged, $c_a(t)$, is simply selected from containers being on top of stacks. The learning process used in methods (D),(E) is as follows:

[1]. The number of containers being on the desired positions is defined as k_B and count k_B

[2]. If $k_B = k$, go to [6] else go to [3],

[3]. Select $c_a(t)$ by using ε-greedy method,

[4]. Select a destination of $c_a(t)$ from the top of stacks by using ε-greedy method,

[5]. Store the state and go to [1],

[6]. Update all the Q-values referred in the episode by eqs. (2),(3).

Since methods (D),(E) do not search explicitly the desired position for each container, each episode is not assured to achieve the desired layout in the early-phase of learning.

In methods (A)-(E), parameters in the yard are set as $k = 18, m_y = n_y = 6$ that are typical values of marshaling environment in real container terminals. Containers are assumed to be loaded in a ship in descendant order from c_{18} to c_1. Figure 5 shows a desired layout for the two cases, and figure 6 shows corresponding initial layout for each case. Other parameters are put as $\alpha = 0.8, \gamma = 0.8, R = 1.0, \varepsilon = 0.8, s = 15$.

The container-movement counts of the best solution and its averaged value for each method are described in Table1. Averaged values are calculated over 20 independent simulations. Among the methods, method (A) derives the best solution with the smallest container-movements. Therefore method (A) can improve the solution for marshaling as well as learning performance to solve the problem.

Results for case 2 are shown in Fig. 7. In the figure, horizontal axis shows the number of trials, and vertical axis shows the minimum number of movements of containers found in the past trials. Each result is averaged over 20 independent simulations. In both cases, solutions that is obtained by methods (A),(B) and (C) is much better as compared to methods (D),(E) in the early-phase of learning, because methods (A),(B),(C) can achieve the desired

Table 1: The best solution of each method for cases 1, 2.

	Case 1		Case 2	
	min. counts	ave. value	min. counts	ave. value
(A)	18	19.10	23	24.40
(B)	20	20.40	25	26.20
Method (C)	34	35.05	35	38.85
(D)	38	46.90	50	64.00
(E)	148	206.4	203	254.0

layout in every trial, whereas methods (D),(E) cannot. Also, methods (A),(B) successfully reduces the number of trials in order to achieve the specific count of container-movements as compared to method (C), since methods (A),(B) considers grouping and finds desirable layouts than can easily diminish the number of movements of container in the early-phase learning. Moreover, at 10000th trail the number of movements of containers in method (A) is smaller as compared to that in method (B) because, among the extended layouts, method (A) obtained better desired layouts for improving the marshaling process as compared to the layout generated by method (B). Desired layouts generated by methods (A),(B) are depicted in the Fig.8 for case 2.

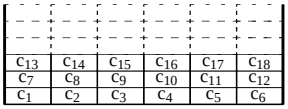


Figure 5: A desired layout for cases 1,2.

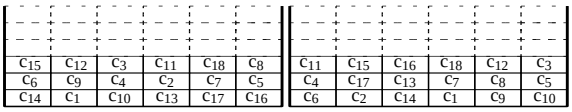


Figure 6: Initial layouts for cases 1,2.

5 CONCLUSIONS

A new reinforcement learning system for marshaling plan at container terminals has been proposed. Each container has several desired positions that are in the same group, and the learning algorithm is designed to considering the feature.

In simulations, the proposed method could find solutions that had smaller number of movements of containers as compared to conventional methods. Moreover, since the proposed method achieves the desired layout in each trial as well as learns the desirable layout, the method can generate solutions with the smaller number of trials as compared to the conventional method.

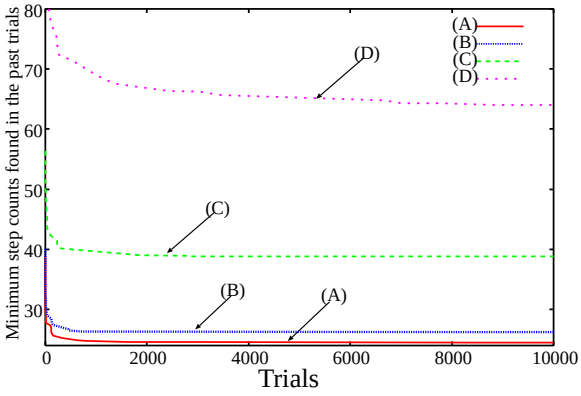


Figure 7: Performance comparison for case 2.

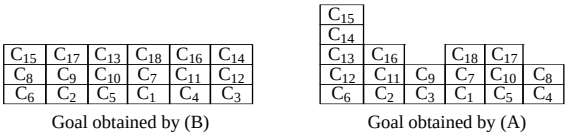


Figure 8: Final layouts of the best solutions for case 2.

REFERENCES

Baum, E. B. (1999). Toward a model of intelligence as an economy of agents. *Machine Learning*, 35:155–185.

Hirashima, Y., Iiguni, Y., Inoue, A., and Masuda, S. (1999). Q-learning algorithm using an adaptive-sized q-table. *Proc. IEEE Conf. Decision and Control*, pages 1599–1604.

Hirashima, Y., takeda, K., Furuya, O., Inoue, A., and Deng, M. (2005). A new method for marshaling plan using a reinforcement learning considering desired layout of containers in terminals. *Preprint of 16th IFAC World Congress*, pages We–E16–TO/2.

Koza, J. R. (1992). *Genetic Programming : On Programming Computers by means of Natural Selection and Genetics*. MIT Press.

Minagawa, M. and Kakazu, Y. (1997). An approach to the block stacking problem by multi agent cooperation. *Trans. Jpn. Soc. Mech. Eng. (in Japanese)*, C-63(608):231–240.

Motoyama, S., Hirashima, Y., Takeda, K., and Inoue, A. (2001). A marshalling plan for container terminals based on reinforce-ment learning. *Proc. of Inter. Sympo. on Advanced Control of Industrial Processes*, pages 631–636.

Siberholz, M. B., Golden, B. L., and Baker, K. (1991). Using simulation to study the impact of work rules on productivity at marine container terminals. *Computers Oper. Res.*, 18(5):433–452.

Watkins, C. J. C. H. and Dayan, P. (1992). Q-learning. *Machine Learning*, 8:279–292.

ON TUNING THE DESIGN OF AN EVOLUTIONARY ALGORITHM FOR MACHINING OPTIMIZATION PROBLEMS

Jean-Louis Vigouroux[1,2], Sebti Foufou[1], Laurent Deshayes[3]
James J. Filliben[4], Lawrence A. Welsch[5] and M. Alkan Donmez[5]
[1]*Laboratoire Electronique, Informatique et Image (Le2i), University of Burgundy, Dijon, France*
[2]*Currently a guest researcher at the National Institute of Standards and Technology (NIST), Gaithersburg, MD, USA*
[3]*Laboratoire de Mécanique et Ingénieries, IFMA and University of Auvergne, Clermont-Ferrand, France*
[4]*Information Technology Laboratory, at NIST, Gaithersburg, MD, USA*
[5]*Manufacturing Engineering Laboratory, at NIST, Gaithersburg, MD, USA*
jean-louis.vigouroux@gadz.org, sfoufou@u-bourgogne.fr, laurent.deshayes@ifma.fr,
james.filliben@nist.gov, lawrence.welsch@nist.gov, alkan.donmez@nist.gov.

Keywords: Evolutionary algorithms, Experimental Algorithmics, Optimization of machining parameters.

Abstract: In this paper, a novel method for tuning the design of an evolutionary algorithm (EA) is presented. The ADT method was built from a practical point of view, for including an EA into a framework for optimizing machining processes under uncertainties. The optimization problem studied, the algorithm, and the computer experiments made using the ADT method are presented with many details, in order to enable the comparison of this method with actual methods available to tune the desing of evolutionary algorithms.

1 INTRODUCTION

A mathematical framework for optimizing machining processes with uncertainties, called the Robust Optimizer, is currently being developed at the National Institute of Standards and Technology (NIST), as a part of the Smart Machining Systems Research Program (Deshayes et al., 2005b). The goal of this framework is to provide a process optimization capability for machine-tools, integrated with the CAD/CAM systems and the machine-tool controllers. Optimization of machining processes is a difficult task which has been studied since the beginning of the last century. As an example, the selection of optimal cutting parameters involves the development of machining models, and optimization algorithms able to handle those models. Taylor built the first experimental models in a seminal study (Taylor, 1907); after him many different models and optimization algorithms were developed. The class of optimization problems encountered can be linear, convex non-linear, or non-convex as shown by Deshayes *et al.* (Deshayes et al., 2005a). For example in high-speed machining processes, the constraint for chatter stability is defined by a non-convex function. Additionally, as shown by Kurdi *et al.* (Kurdi et al., 2005), uncertainties on model parameters may strongly influence the limit contours of the constraint. As a consequence, the framework under development must contain several algorithms for optimization under uncertainties such as robust linear programming, stochastic programming, depending on the class of optimization problems. Vigouroux *et al.* (Vigouroux et al., 2007) presented a novel optimization algorithm to fit into the framework, for non-convex problems, by coupling an evolutionary algorithm with interval analysis. EA are algorithms from the Artificial Intelligence domain. The main mechanism of EA is an iterative sampling of candidate solutions to find the optimal one, for more information see the book of Yao (Yao, 1999). The latest applications of EA include Pattern recognition (Raymer et al., 2000), multi-disciplinary optimization for car design (Rudenko et al., 2002). EA have already been applied in machining by Wang *et al.* (Wang et al., 2002) to a non-convex optimization problem without uncertainties. EA are algorithms that require tuning of their parameters to obtain optimal results regarding execution time, convergence, and accuracy. Moreover the parameters must be robust for a range of similar problems.

The design of evolutionary algorithms, that is the choice of an EA architecture and algorithm parameters, robust to problem parameter changes, or adapted to several problems is not straightforward. There-

fore, evolutionary algorithms have found applications in specific optimization problems where the tuning of the algorithm can be performed by the user. Several methods to help this design were introduced recently by Francois and Lavergne (Francois and Lavergne, 2001), Nannen and Eiben (Nannen and Eiben, 2006). Nannen and Eiben were interested to design an evolutionary system to study evolutionary economics. For that complex system, EA are not just an optimization algorithm but also a model for simulating evolution. Therefore the EA used by Nannen and Eiben contain much more parameters than the EA studied here: over ten. Nannen and Eiben don't have an alternate method at hand to find the optimum solution, mainly because the EA studied does not perform optimization. To optimize the parameters of the EA, they built the CRE method, by considering a trade-off between a performance index,the average response of the EA, and the average algorithm complexity, for 150 samples of the algorithm parameters. To take in account similar problems, they let vary three problem parameters and do 18 experiments. For each of them, an iterative routine determines automatically the optimal values of the algorithm parameters, based on the trade-off. Then the optimal values of the parameters are found as the average over all experiments.

The relevance of such methods is important to integrate EA in the Robust Optimizer. Machining problems have many parameters subject to changes. In this paper, an alternative method to the CRE method, the algorithm design tuning (ADT) method, is presented for helping the design of an EA for machining optimization problems. It makes use of statistical engineering tools not only to optimize the algorithm parameters, but to modify the EA design in order to avoid non-convergence. In the background of the study are presented the example turning problem used for building the Robust Optimizer, along with the EA able to solve it, and some notions about experimenting with algorithms. Then the novel ADT method is introduced along with experimental results and a comparison with the CRE method.

2 BACKGROUND

2.1 Example Turning Problem for the Robust Optimizer

An optimization problem consists of optimizing an objective function while satisfying several constraints:

$$\text{Minimize } \mathbf{f}(\mathbf{x})$$

$$\text{subject to } g_j(\mathbf{x}) \leq 0 \quad j = 1,2 \ldots p$$

where **f** is the objective function, g_j a constraint function, and **x** is a vector of real variables, called a solution of the problem. The vectors $\mathbf{x}^l$ and $\mathbf{x}^u$ define the lower and upper bounds for the variables, and are specified in some of the constraint equations g_j. These bounds define the search space.

The turning problem is a two-variable optim ization problem, with a set of specific values for the problem parameters. The variables and parameters used in the problem are presented below:
f- Feed, mm/rev
Vc- Cutting speed, m/s
a_p- Depth of cut, mm
f_{min}- Minimum Feed, mm/rev
f_{max}- Maximum Feed, mm/rev
Vc_{min}- Minimum Cutting speed, m/s
Vc_{max}- Maximum Cutting speed, m/s
Pu- Spindle power, W
Cu- Spindle torque, $N\dot{m}$
Ra_{spe}- Workpiece desired surface roughness, μm
Ra- Workpiece surface roughness, μm
K_1,$\delta r_{epsilon}$- Constants in surface roughness equation
$r_{epsilon}$- Tool nose radius, mm
F_c- Cutting force, N
K_c- Constant in cutting force equation
tc- Exponent in cutting force equation
T- Tool life, s
T_r, f_r, a_{pr}, Vc_r- Constants in tool life equation
n,m,l- Exponents in tool life equation
t_{cut}- Cutting time, s
N_{parts}- Desired number of parts to cut per tool
A- Workpiece equivalent cutting surface, mm^2
D_{av}- Workpiece average diameter, mm
L_c- Workpiece equivalent length of cut, mm

The goal is to find the optimal machining parameters, feed and speed, which maximize the material removal rate (MRR) of one turning pass, considering several process constraints. The two variables to be optimized are the feed f and the cutting speed Vc. The mathematical functions of the problem are detailed below:
The objective function is defined as the opposite of MRR in order to solve a minimization problem[1]:

$$\mathbf{f}(f,Vc) = -ap \times f \times Vc \tag{1}$$

The constraint function of minimum feed, C1:

$$g_1(f,Vc) = f_{min} - f \tag{2}$$

The constraint function of maximum feed, C2:

$$g_2(f,Vc) = f - f_{max} \tag{3}$$

[1]To maximize an objective function is the same as to minimize its negative opposite.

The constraint function of minimum cutting speed, C3:

$$g_3(f,Vc) = Vc_{min} - Vc \quad (4)$$

The constraint function of maximum cutting speed, C4:

$$g_4(f,Vc) = V_c - Vc_{max} \quad (5)$$

The constraint function of surface roughness, C5:

$$g_5(f,Vc) = Ra(f) - Ra_{spe} \quad (6)$$

The constraint function of tool life, C6:

$$g_6(f,Vc) = T(f,Vc) - t_{cut}(f,Vc) \quad (7)$$

The constraint function of Spidle power life, C7:

$$g_7(f,Vc) = Pu - F_c(f,Vc) \times V_c \quad (8)$$

Several functions have variables defined by the equations presented below. The tool life equation is obtained from Taylor's work (Taylor, 1907):

$$T(f,Vc) = T_r \frac{a_p}{a_{pr}}^{l} \frac{f}{f_r}^{m} \cdot \frac{V_c}{V_{cr}}^{n} \quad (9)$$

The cutting time, t_{cut}, is obtained from the following equation:

$$t_{cut}(f,Vc) = N_{parts} \frac{A}{f \times V_c \times 1000}, \quad (10)$$

where A is obtained from the following equation:

$$A = \pi \times D_{av} \times L_c. \quad (11)$$

The surface roughness equation is obtained from the study by Deshayes *et al.* (Deshayes et al., 2005a):

$$Ra(f) = K_1 f \frac{\delta r_{epsilon}}{r_{epsilon}}. \quad (12)$$

The cutting force equation is obtained from an experimental study by Ivester *et al.* (Ivester et al., 2006):

$$Fc(f) = Kc \times f^{1+tc} \times ap. \quad (13)$$

The original problem has parameter values given in Table (1). For this original problem, the limit contours of the constraint functions are represented in Figure (1). Ten parameters of this problem can be modified and are specified in Table (1). The parameter ranges of this table are determined by knowledge of machining and judgement.

Table 1: Problem parameter ranges.

Parameter name	Original value	Minimum value	Maximum value	Factor name
Vc_{min}	1.83	1.33	2.5	X1
Vc_{max}	7.5	6.5	10	X2
f_{min}	0.2	0.1	0.3	X3
f_{max}	0.6	0.5	0.8	X4
a_p	2	1	4	X5
K_c	2000	1800	2500	X6
Pu	15000	15000	20000	X7
D_{av}	72	60	80	X8
L_c	550	350	650	X9
Ra_{spe}	4.5	3	5	X10

2.2 Evolutionary Algorithms

In Darwin's evolution theory (Darwin, 1859), a population of individuals evolves over several generations. The individuals with the best fitness, determined by their genes, are selected to breed; their children form the next population. Since offspring of individuals with best fitness should be better suited to survive within their environment, the species improves. Thus, over time and several generations, evolution leads to the adaptation of the population to the environment.

This theory is used by the implemented Evolutionary Algorithm (EA). The optimization variables play the role of the individuals' genes. The EA is presented as a class written in the Python language[2]. The constructor function of this class is presented below:

```
class EA:
  def __init__(self):
    self.gene_size=2
    x1bounds=[0.2,0.6]
    x2bounds=[1.83,7.5]
    self.bounds = [x1bounds,x2bounds]
    self.N = 50
    self.alpha=0.4
    self.z=20
```

The constructor sets the number of genes per individual (here two), the bounds of variables, the population size N, the breeding parameter α, and the number of generations z. The `run_algorithm` function contains the algorithm steps and is presented here:

```
def run_algorithm():
  population = population_initialize()
  for i in range(self.z):
    parent_population =
    tournament_selection(population)
    children_population =
    breeding(parent_population)
    population = children_population
  return population[1]
```

[2] *Certain trade names and company products are mentioned in the text or identified in an illustration in order to adequately specify the experimental procedure and equipment used. In no case does such an identification imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the products are necessarily the best available for the purpose. Official contribution of the National Institute of Standards and Technology; not subject to copyright in the United States.*

The EA has the following steps:

- Construction of the initial population. The `population_initialize` function, presented below, outputs a first population of N individuals spread uniformly over the search space, by use of a `random` function giving a number between zero and one. The fitness function, *h*, of one individual is computed with the static penalty method, and is given in Equation (14). Michalewicz *et al.* (Michalewicz et al., 1996) reviewed several methods that enable EA to handle optimization problems with constraints, and explain this method with more details.

```
def population_initialize():
  population = []
  for i in range(self.pop_size):
    population.append([])
    for j in range(self.gene_size):
      population[i].append(self.bounds[j][0]+
      (self.bounds[j][1]
      - self.bounds[j][0])*random())
      population[i].append(fitness(population[i]))
  return population
def fitness(genes):
  result=h(genes)
  return result
```

$$h(\mathbf{x}) = -sf(\mathbf{x}) + W \times \sum_{j=1}^{n} sg_j(x) \qquad (14)$$

with

$$sf(x) = \frac{\mathbf{f}(\mathbf{x})}{\mathbf{f}(\mathbf{x^u})} \qquad (15)$$

and

$$sg_j(\mathbf{x}) = \frac{g_j(\mathbf{x})}{g_j(\mathbf{x^u})} \quad if \quad g_j(\mathbf{x}) \leq 0, \quad 0 \quad else \qquad (16)$$

W is the static penalty parameter. This parameter controls the degradation of the fitness function when one or more constraints are violated. In this article, the value of W is ten.

- Selection. The `tournament_selection` function, presented below, outputs a parent population of N/2 individuals. These individuals are selected by N/2 tournaments of size 2: for each tournament two individuals are randomly picked in the population and their fitness functions are compared. The individual with the higher fitness value wins and is added to the parent population. The loser re-enters the population and can eventually participate in another tournament.

```
def tournament_selection(population):
  pool_size=self.N/2
  parent_population = []
  for i in range(pool_size):
    parent_population.append([])
    match = Structure()
    match.pool = sample(population,2)
    match.knight1=match.pool[0]
    match.knight2=match.pool[1]
    if match.knight1[2] <match.knight2[2]:
      match.winner = match.knight1
    elif match.knight1[2] ==match.knight2[2]:
       match.winner = match.knight1
    else:
       match.winner = match.knight2
       parent_population[i] = match.winner
  return parent_population
```

- Breeding. The `breeding` function, presented below, outputs a new children population. Two individuals are randomly picked from the parent population. Two new individuals are created in a box. The position of this box within the search space is defined by the two parents' genes. The dimensions of the box are defined by both a breeding parameter called α and the two parents' genes. This process of creating individuals is called blend crossover and was introduced by Eschelman and Schaffer (Eschelman and Schaffer, 1993). It is repeated N/2 times in order to produce a children population having N individuals.

```
def breeding(parent_population):
  children_population = []
  for i in range(self.pool_size):
    parents = sample(parent_population,2)
    parent1 = parents[0]
    parent2 = parents[1]
    children1 = []
    children2 = []
    for j in range(self.gene_size):
      a= -self.alpha +(1+2*self.alpha)*random()
      children1.append(a*parent1[j] +
                              (1-a)*parent2[j])
      a= -self.alpha +(1+2*self.alpha)*random()
      children2.append(a*parent1[j] +
                              (1-a)*parent2[j])
      children1.append(fitness(children1))
      children2.append(fitness(children2))
      children_population.append(children1)
      children_population.append(children2)
    return children_population
```

At the next generation, this new population will face again the selection step and the loop will continue until the number of generations *z*, is attained. When the algorithm is stopped, the first individual in the population, with genes (x_{11},x_{21}), is picked. The objective function MRR_z is computed for this individual in Equation (17) and is the result output by the algorithm.

$$MRR_z = a_p \times x_{11} \times x_{21} \qquad (17)$$

For the original problem, the populations at several generations are plotted in Figure (1). The initial population is represented by stars, the population at generation 2 is represented by circles, and the population at generation 5 is represented by diamonds. The

evolution of the center of gravity of the population over several generations defines a track represented by crosses and dashed lines. This algorithm has three influential parameters: the breeding parameter α, the population size N, and the number of generations, z. Tuning of these parameters is required to enhance the performance of the algorithm.

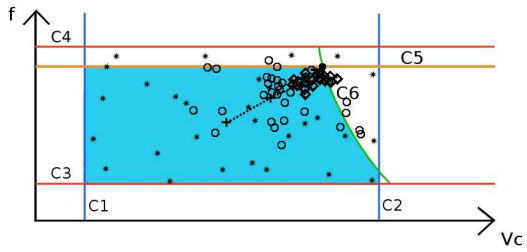


Figure 1: Populations at several generations during one run of the EA.

2.3 Experimental Algorithmics

Experimental Algorithmics refer to the study of algorithms through a combination of experiments and classical analysis and design techniques.

Since EA are randomized algorithms, their results are unequal run after run, even though all parameters may remain the same. Therefore, it is necessary to run the EA several times to report an average result. Thus, the computational expense associated with the tuning of the algorithm may become important. Statistical techniques such as design of experiments and response surface methodology avoid this computational expense and are presented in (NIST/SEMATECH, 2006) and (Box et al., 1978).

A computer experiment refers here to a run of the EA with a set of algorithm parameters and a set of problem parameters. With the design of experiments, the problem parameters are renamed factors, see Table (1); a design stands for a set of experiments, with each experiment having a specific combination of factor levels. When considering only two levels of variation for each problem parameter, the low settings and the high settings presented in Table (1), there are 1024 possible combinations. A full factorial design is a set with all possible factor levels combinations, here $2^{10} = 1024$. A more efficient design is an orthogonal fractional factorial design (NIST/SEMATECH, 2006), with only 128 runs, ten percent of the full factorial design.

According to the response surface terminology (Box et al., 1978), the measures made during the experiments are referred to as response variables.

For the design of the EA, three response variables are considered:

- y_1 is an index defining the accuracy of the EA:

$$y_1 = \frac{MRR_z}{MRR^*} \tag{18}$$

  where MRR^* is the optimal material removal rate of the problem, corresponding to the optimal solution. MRR^* can be obtained graphically or by another optimization algorithm able to find it. Since MRR is an increasing function, The expected range of y_1 is between 0 and 1. If the value of y_1 is greater, the result MRR_z output by the algorithm is unacceptable and it means that the algorithm did not converge.

- y_2 is the count of non-convergence cases after several runs of the algorithm. Those non-convergence cases are detected by values of y_1 higher than 1.

- y_3 is a time measure in seconds, related to the execution time of the EA.

The set-up of the computer experiments is given: The EA is implemented with Matlab software on a personal computer. The computer processor is an Intel Pentium 4 with a frequency of 3.80 GHz, running with a memory of size 2 GB. The measure of y_3 is made using Matlab's **cputime** function.

The random function used to generate the necessary random behaviors in the algorithm is Matlab's **random** function with the **Seed** method. It generates a pseudo-random number between 0 and 1. After a certain number of calls, $2^{31} - 1$, the function repeats itself. This function takes as the argument a number called the seed. The seed defines the internal state of the function and, with the same seed, the random function generates the same sequence of random numbers. Therefore, all the seeds used in the experiments were recorded and stored, giving the possibility to reproduce exactly the same experimental results. Each computer experiment was replicated ten times with different seeds.

Once the computer experiments have been made, the Exploratory Data Analysis (EDA) techniques (Tukey, 1977) can be used to explore the results. These techniques allow to ferret out interesting attributes of the result data. The Dataplot statistical engineering software was used in this study to perform EDA. Sensitivity analysis is a technique that allows to order the problem parameters according to the effect of their variation on the response. The order of the parameter can be deduced from a plot called the main effect plot. An example plot is shown in Figure (3).

3 THE ADT METHOD

3.1 Step 1: Preliminary Exploratory Data Analysis

The first step of the methodology is to perform an exploratory analysis. The goal is to see how much the algorithm performances vary, and to establish if the algorithm needs any modifications of its architecture. This analysis requires performing computer experiments with a set of algorithm parameters (α,N,z), according to the orthogonal fractional factorial design introduced previously. This step requires a special plot of the response y_1: the responses of all experiments are ordered and plotted, as shown in Figure (2). The experiments where the algorithm does not converge are detected by values of y_1 exceeding one. The reason for non-convergence can be discovered by plotting the limit contours of the constraints for the case with non convergence, similarly to Figure (1) presented previously. Then the algorithm architecture may be modified, in order to be adapted to the non-convergence cases.

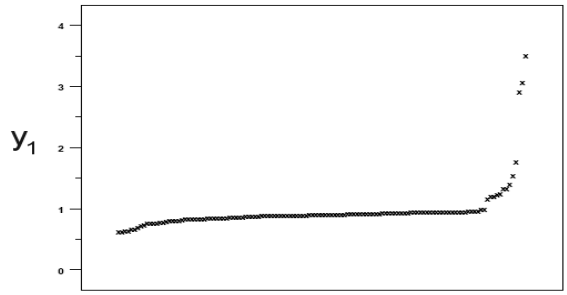


Figure 2: Ordered data plot of response y_1 for the exploratory analysis of the set ($\alpha = 0.4$, $N = 40$,$z = 20$).

3.2 Step 2: Sensitivity Analysis

This step is optional. This step may only be required if the problem has more than two optimization variables, where it may be impossible to plot the limit contours of the constraints for the case with non-convergence. In that case the reason for non-convergence cannot be easily discovered. The sensitivity analysis allows to determine the most important problem parameters affecting the response y_1. The main effect plot in Figure (3) shows the variation of the response y_1 depending on the variation of the problem parameters.

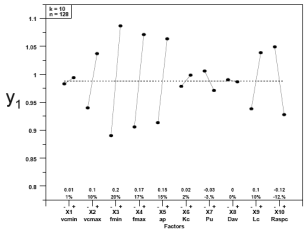


Figure 3: Main effect plot of response y_1 for the sensitivity analysis of the set ($\alpha = 0.4$, $N = 40$,$z = 20$).

3.3 Step 3: Optimization

In this step the algorithm parameters are optimized(α,N,z). α,N,z can be chosen respectively in the range [20, 200],[0.1, 0.9] and [10, 50]. Given a set (α,N,z), a sensitivity analysis is performed as previously discussed, but the results of each computer experiment are now three responses, y_1, y_2, and y_3. The goal is to optimize the average values of the responses: maximize $\bar{y_1}$, minimize $\bar{y_2}$, minimize $\bar{y_3}$.

First, only the first objective $\bar{y_1}$ is considered. Since the computational expense of the sensitivity analysis necessary to compute $\bar{y_1}$ is high, only a small number of sets (α,N,z) can be investigated. The response surface methodology is applied to find the best set (α,N,z). Starting with an initial guess (α_0,N_0,z_0), several other sets in the neighborhood are tested, whose values are defined by an orthogonal full factorial design. This design has 3 factors and 11 experiments. The choice of the parameter ranges, centered around the set (α_0,N_0,z_0), is left to the user.

The result $\bar{y_1}$ of each tested set (α,N,z) can be visualized as a cube, as shown in Figure (4). From the analysis of the cube, a new center set (α_1,N_1,z_1) is chosen, to explore the space in the region where the optimum value of $\bar{y_1}$, $\bar{y_1}^*$, could be located. Again, computer experiments are performed with the same design but with different parameter ranges, in order to acquire the new cube with center (α_1,N_1,z_1). This iterative process can be repeated as long as needed in order to find $\bar{y_1}^*$. The cubes for the objectives $\bar{y_2}$ and $\bar{y_3}$ can be acquired at the same time, although the optimum values for these objectives are not found. This method enables one to explore the spaces $\bar{y_2}$ and $\bar{y_3}$ as functions of variables (α,N,z). After the three objective spaces have been explored, the user is able to pick a set (α,N,z), that has good values $\bar{y_1}$, $\bar{y_2}$, and $\bar{y_3}$, although not optimal. This set represents a compromise between best accuracy, best convergence, and best execution time.

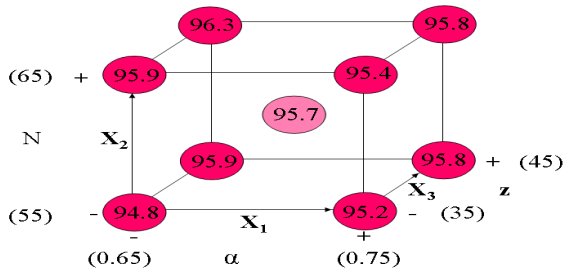


Figure 4: The responses $\bar{y_1}$ multiplied by 100 for the cube with center ($\alpha_0 = 0.7$, $N_0 = 60$, $z_0 = 40$).

3.4 Validation of the Optimized Algorithm

The algorithm is optimized with a sample of problem parameter combinations and a sample of seeds. This sample can be termed a training data set, and an analogy can be drawn with the training of neural networks. Now the performance results of the algorithm must be verified with a new sample of data: the validation data set.

In order to compare the average performance results, a validation data set of the same size as the training data set must be chosen. A different sampling of the problem parameter combinations is made. A Latin hypercube sampling is used with the same size as the orthogonal design presented in Section (2.3): 128 values for each of the ten factors. A new sample of 100 different seeds is also picked and stored. The sensitivity analysis can be replicated ten times with the new data and the results compared.

3.5 Results

The preliminary exploratory analysis was performed on the set ($\alpha = 0.4$, $N = 40$, $z = 20$). The ordered data plot of response y_1, shown in Figure (2), revealed non-convergence cases, detected on the right of the plot by values exceeding one. The limit contours of the constraints were plotted for some of the non-convergence cases, and it was discovered that the initialization step was not adapted. To avoid this problem, it was decided to change the initialization so that the first population is spread in the lower left quarter of the search space. The sensitivity analysis, shown in Figure (3), was performed, but all the information provided by this step was already known from the plots of the first step.

In the third step, optimization was performed with an initial guess ($\alpha_0 = 0.7$, $N_0 = 60$, $z_0 = 40$). After three iterations, it was found that $\bar{y_1}^*$ should have a value around 0.97, and a good confidence was acquired about the knowledge of the three objective spaces. The set of algorithm parameters selected is ($\alpha = 0.6$, $N = 60$, $z = 40$). The results of a sensitivity analysis of this set, replicated ten times, is shown in Table (2). During the validation step, a comparison was made between the results obtained with the training data set and the results obtained with the validation data set. The later results presented in Table (3) are slightly better. Therefore, the training method used to tune the algorithm seems to be sound.

Table 2: Optimization step: results for the set ($\alpha = 0.6$,$N = 60$,$z = 40$) with the training data set.

	Response variable $\bar{y_1}$	Response variable $\bar{y_2}$	Response variable $\bar{y_3}$ (s)
average	0.950	1	1.130
standard deviation	0.002	1.05	0.003

Table 3: Validation step: results for the set ($\alpha = 0.6$,$N = 60$,$z = 40$) with the validation data set.

	Response variable $\bar{y_1}$	Response variable $\bar{y_2}$	Response variable $\bar{y_3}$ (s)
average	0.966	0	1.123
standard deviation	0.001	0.00	0.002

3.6 Comparison with the Cre Method

The ADT method was developped for the turning problem of the Robust Optimizer. From this point of view the EA has few parameters, three, and quite a lot of problem parameters, ten. On the contrary, the CRE method considers many EA parameters and few problem parameters. The reason why the CRE method does not consider many problem parameters may be a reason of time consumed to perform the experiments, as indicated in the paper by Nannen and Eiben (Nannen and Eiben, 2006). The ADT method required also a lot of time to perform the experiments. For both methods, increasing the number of algorithm parameters and problem parameters considered is a problem, so the complexity of the CRE and ADT methods should be determined and compared.

The CRE method relies on information theory to determine the importance of the algorithm parameters. This point is not adressed by the ADT method. During the study, it was decided to remove the mutation parameter from the algorithm by studying videos showing the populations evolve over the search space, during several iterations. This kind of decision was made possible because of the low dimension of the optimization problem studied: only two variables.

The ADT method enables to optimize very practical responses of EA: the execution time, the number of non-convergence cases, the accuracy. For the last objective, accuracy, one needs to know the optimal solution for each optimization problem. This is possible graphically only for low dimension optimization problems. For optimization problems of higher dimension, an alternative optimization algorithm must be used.

Since this method was developped concurrently with the CRE method, an experimental comparison of the two methods was not possible. However all details are given in this paper to perform such a study.

4 CONCLUSION

The novel ADT method presented here enables tuning the design of an EA for optimizing a turning process with uncertainties. The ADT method also enables the algorithm designer to manage several performance objectives. The method is an alternative method to the CRE method, and focuses on the practical objectives of accuracy, non-convergence and execution time. In the future, the two methods need to be compared. For the Robust Optimizer, an optimization problem involving a milling process needs to be considered, to see if the ADT method enables to design an EA for optimizing different kinds of machining processes, or not.

ACKNOWLEDGEMENTS

The authors would like to acknowledge the support and assistance of Robert Ivester, David Gilsinn, Florian Potra, Shawn Moylan, and Robert Polvani, from the National Institute of Standards and Technology.

REFERENCES

Box, G. E. P., Hunter, W. G., and Hunter, J. S. (1978). *Statistics for Experimenters*. John Wiley.

Darwin, C. (1859). *On the origin of species by means of natural selection*. John Murray.

Deshayes, L., Welsch, L., Donmez, A., and Ivester, R. W. (2005a). Robust optimization for smart machining systems: an enabler for agile manufacturing. In *Proceedings of IMECE2005: 2005 ASME International Mechanical Engineering Congress & Exposition*.

Deshayes, L., Welsch, L., Donmez, A., Ivester, R. W., Gilsinn, D., Rohrer, R., Whitenton, E., and Potra, F. (April 3-5, 2005b). Smart machining systems: Issues and research trends. In *Proceedings of the 12th CIRP Life Cycle Engineering Seminar*.

Eschelman, L. and Schaffer, J. D. (1993). *Real-coded genetic algorithms and interval-schemata*. Morgan Kaufmann.

Francois, O. and Lavergne, C. (2001). Design of evolutionary algorithms-a statistical perspective. *IEEE Transactions on Evolutionary Computation*, 5(2):129–148.

Ivester, R. W., Deshayes, L., and McGlauflin, M. (2006). Determination of parametric uncertainties for regression-based modeling of turning operations. *Transactions of NAMRI/SME*, 34.

Kurdi, M. H., Haftka, R. T., Schmitz, T. L., and Mann, B. P. (November 5-11, 2005). A numerical study of uncertainty in stability and surface location error in high-speed milling. In *IMECE '05: Proceedings of the IMECE 2005*, number 80875. ASME.

Michalewicz, Z., Dasgupta, D., Riche, R. L., and Schoenauer, M. (1996). Evolutionary algorithms for constrained engineering problems. *Computers & Industrial Engineering Journal*, 30(2):851–870.

Nannen, V. and Eiben, A. (2006). A method for parameter calibration and relevance estimation in evolutionary algorithms. In *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 183–190, New York, NY, USA. ACM Press.

NIST/SEMATECH (2006). Nist/sematech engineering statistics handbook, http://www.itl.nist.gov/div898/handbook/.

Raymer, M., Punch, W., Goodman, E., Kuhn, L., and Jain, A. (2000). Dimensionality reduction using genetic algorithms. *IEEE Trans. Evolutionary Computation*, 4:164–171.

Rudenko, O., Schoenauer, M., Bosio, T., and Fontana, R. (Jan 2002). A multiobjective evolutionary algorithm for car front end design. *Lecture Notes in Computer Science*, 2310.

Taylor, F. N. (1907). On the art of cutting metals. *Transactions of the ASME*, 28:31.

Tukey, J. W. (1977). *Exploratory data analysis*. Addison-Wesley.

Vigouroux, J., Deshayes, L., Foufou, S., and Welsch, L. (2007). An approach for optimization of machining parameters under uncertainties using intervals and evolutionary algorithms. In *Proceedings of the first International Conference on Smart Machining Systems, to appear*.

Wang, X., Da, Z. J., Balaji, A. K., and Jawahir, I. S. (2002). Performance-based optimal selection of cutting conditions and cutting tools in multipass turning operations using genetic algorithms. *International Journal of Production Research*, 40(9):2053–2059.

Yao, X. (1999). *Evolutionary Computation: Theory and Applications*. World Scientific.

TRACKING CONTROL DESIGN FOR A CLASS OF AFFINE MIMO TAKAGI-SUGENO MODELS

Carlos Ariño
Department of Systems Engineering and Design, Jaume I University, Sos Baynat S/N, Castelló de la Plana, Spain
arino@esid.uji.es

Antonio Sala, Jose Luis Navarro
Department of Systems Engineering and Control, Universidad Politcnica de Valencia, Camino de Vera, 14, Valencia, Spain
asala@isa.upv.es, jlnavarro@isa.upv.es

Keywords: Fuzzy control, affine Takagi-Sugeno models, local models, LMI, setpoint change.

Abstract: When controlling Takagi-Sugeno fuzzy systems, verification of some sector conditions is usually assumed. However, setpoint changes may alter the sector bounds. Alternatively, setpoint changes may be considered as an offset addition in many cases, and hence affine Takagi-Sugeno models may be better suited to this problem. This work discusses a nonconstant change of variable in order to carry out offset-ellimination in a class of MIMO canonical affine Takagi-Sugeno models. Once the offset is cancelled, standard fuzzy control design techniques can be applied for arbitrary setpoints. The canonical models studied use as state representation a set of basic variables and their derivatives. Some examples are included to illustrate the procedure.

1 INTRODUCTION

In the last decade, design of fuzzy controllers based on the so-called Takagi-Sugeno TS models (Takagi and Sugeno, 1985) has reached maturity (Sala et al., 2005). TS models express the behaviour of a system via a convex interpolation of local (homogeneous) linear models, where the interpolation functions are fuzzy membership functions with and add-1 conditions. In particular, designs using the Linear Matrix Inequality framework (Tanaka and Wang, 2001; Guerra and Vermeiren, 2004) have become widespread. Part of the success of such techniques is due to the existence of systematic methodologies for TS fuzzy identification (Takagi and Sugeno, 1985; Tanaka and Wang, 2001; Babuska, 1998; Nelles et al., 2000).

One particular characteristic of the mainstream TS control design framework is that all of the local models must share the same equilibrium point, usually set to $x = 0$ for convenience. This is not a severe problem, as the identification procedures above referred need only be applied with a constant change of variable, used in the context of Taylor linearisation in control design for decades. Once that change of variable is carried out, global stability and performance regarding reaching $x = 0$ from any initial conditions can be proved. The reader is referred to (Tanaka and Wang, 2001; Guerra and Vermeiren, 2004) for details on the methodology.

An affine structure for TS models was also originally addressed in (Takagi and Sugeno, 1985), which considers local models without the shared equilibrium point. This structure, to be denoted as Takagi-Sugeno-Offset (TSO) may originate either directly from the identification process or when considering tracking tasks with varying setpoints in ordinary TS models. Indeed, in the latter case, the change of variable needed to transform the new operating point into $x = 0$ should involve changing the shape of the membership functions and the parameters of the local models. Otherwise, the resulting models lose the shared equilibrium point.

The above mentioned control methodologies must be adapted to TSO models. Some ideas appear in (Kim and Kim, 2002; Johansson, 1999), where quadratic Lyapunov functions and S-procedure LMIs (Boyd et al., 1994) are used to prove stability of the origin. However, setpoint changes are not considered, and the division into ellipsoidal zones of the operating regime results in a cumbersome procedure which involves considering the different regions of overlap of the antecedent membership functions.

As an alternative, this paper presents a particular

class of TSO fuzzy models stemming from a canonical representation which, basically, takes the controlled outputs and its derivatives as the chosen set of state variables. This canonical representation has a clear physical insight and stems from usual canonical forms in linear and nonlinear systems (Antsaklis Panos and Michel Anthony, 1997; Slotine and Li, 1991).

Once the canonical TSO models are introduced, an offset-removing transformation for state feedback is discussed, which is the main result of this work. The offset-removing transformation will allow to express the original TSO model as an ordinary TS form on the transformed variables, enabling standard fuzzy TS control design techniques to be used in such a system.

The structure of the paper is as follows. Section 2 will present the definitions for the canonical TSO framework. Section 3 will present results on the equilibrium point of the canonical representation and the main offset-removing transformation. Some examples illustrating the approach will be given in Section 4, and a conclusion section will close the paper.

2 CANONICAL TSO MODELS

In this section, basic definitions of the fuzzy systems under study will be presented, which generalise the classical Takagi-Sugeno fuzzy system used in most current design techniques (Tanaka and Wang, 2001; Guerra and Vermeiren, 2004; Sala et al., 2005) described by

$$\dot{x} = \sum_{i=1}^{n} \mu_i(z)(A_i x + B_i u) \quad \sum_i \mu_i(z) = 1 \qquad (1)$$

where z is assumed to be a set of accessible variables which may include some or all of those ones comprising the state vector x plus external scheduling ones, and $\mu_i(z)$ are denoted as antecedent membership functions.

Definition 1 *Canonical local model with offset*
Let us have a system with p inputs, p outputs and n states defined by:

$$\begin{aligned} \dot{x} &= A \cdot x + B \cdot u + R \\ y &= C \cdot x \end{aligned} \qquad (2)$$

where the state vector is assumed to be partitioned according to the following structure:

$$x = [x_{11}\ x_{12} \ldots x_{1r_1}\ x_{21}\ x_{22} \ldots x_{2r_2}\ \ldots\ x_{p1}\ x_{p2} \ldots x_{pr_p}]^T \qquad (3)$$

i.e., *p blocks of size $r_1, \ldots, r_p$ respectively, $r_1 + \cdots + r_p = n$, compatible with the block structure in matrices A, B, C, R to be described below.*

Let us define an auxiliary matrix with dimension $q \times (q-1)$ as:

$$T_q = [0_{(q-1)\times 1}\ I_{q-1}] \qquad (4)$$

where I_{q-1} denotes the identity matrix with size $(q-1) \times (q-1)$ and $0_{(q-1)\times 1}$ the zero matrix with size $(q-1) \times 1$. Also, the notation $[\ ]_{s\times t}$ will denote a matrix with dimension $s \times t$ with arbitrary elements. Then, matrices in (2) *have the structure:*

$$A = \left[\begin{array}{c|c|c|c} T_{r_1} & 0_{(r_1-1)\times r_2} & \cdots & 0_{(r_1-1)\times r_p} \\ \hline \multicolumn{4}{c}{[\ ]_{1\times n}} \\ \hline 0_{(r_2-1)\times r_1} & T_{r_2} & \cdots & 0_{(r_2-1)\times r_p} \\ \hline \multicolumn{4}{c}{[\ ]_{1\times n}} \\ \hline \multicolumn{4}{c}{\cdots} \\ \hline 0_{(r_p-1)\times r_1} & \cdots & 0_{(r_p-1)\times r_{p-1}} & T_{r_1} \\ \hline \multicolumn{4}{c}{[\ ]_{1\times n}} \end{array}\right] \qquad (5)$$

$$B = \left[\begin{array}{c} 0_{(r_1-1)\times p} \\ \hline [\ ]_{1\times p} \\ \hline 0_{(r_2-1)\times p} \\ \hline [\ ]_{1\times p} \\ \hline \vdots \\ \hline 0_{(r_p-1)\times p} \\ \hline [\ ]_{1\times p} \end{array}\right] \qquad (6)$$

$$C = \left[\begin{array}{c|c|c|c|c|c|c} 1 & & 0 & & & 0 & \\ 0 & & 1 & & & 0 & \\ 0 & & 0 & & & \vdots & \\ \vdots & [\ ]_{p\times(r_1-1)} & \vdots & [\ ]_{p\times(r_2-1)} & \cdots & 0 & [\ ]_{p\times(r_p-1)} \\ 0 & & 0 & & & 1 & \end{array}\right] \qquad (7)$$

$$R = \left[\begin{array}{c} 0_{(r_1-1)\times p} \\ \hline [\ ]_{1\times p} \\ \hline 0_{(r_2-1)\times p} \\ \hline [\ ]_{1\times p} \\ \hline \vdots \\ \hline 0_{(r_p-1)\times p} \\ \hline [\ ]_{1\times p} \end{array}\right] \qquad (8)$$

Note 1 *The above system structure is similar to the well-known reachable canonical form (Antsaklis Panos and Michel Anthony, 1997). For instance, a canonical SISO system:*

$$A = \begin{bmatrix} 0 & 1 & 0 & \ldots & 0 \\ 0 & 0 & 1 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & 0 & \ldots & 1 \\ -a_1 & -a_2 & -a_3 & \ldots & -a_q \end{bmatrix} \qquad (9)$$

$$B = \begin{bmatrix} 0 & 0 & 0 & \ldots & b \end{bmatrix}^T \qquad (10)$$

$$C = \begin{bmatrix} 1 & c_2 & c_3 & \dots & c_q \end{bmatrix} \quad (11)$$

$$R = \begin{bmatrix} 0 & 0 & 0 & \dots & r \end{bmatrix}^T \quad (12)$$

conforms to the above structure.

Definition 2 *Canonical Fuzzy Takagi-Sugeno-Offset model.*
A Fuzzy canonical Takagi-Sugeno-Offset model will be defined according to the following structure:

$$\begin{aligned} \dot{x} &= \sum_i^m \mu_i(z)(A_i \cdot x + B_i \cdot u + R_i) \\ y &= \sum_i^m \mu_i(z) C_i x \end{aligned} \quad (13)$$

where each of the component models has matrices A_i, B_i, R_i y C_i which follow the structure in Definition 1 and $\mu_i(z)$ are the membership functions, which are assumed to verify $\sum_i \mu_i(z) = 1$.

The notation below will be used as shorthand for fuzzy summations

$$\widetilde{\Omega}(z) = \sum_i \mu_i(z)\Omega_i \quad (14)$$

Then, the fuzzy system in Definition 2 may be written as:

$$\begin{aligned} \dot{x} &= \widetilde{A}(z) \cdot x + \widetilde{B}(z) \cdot u + \widetilde{R}(z) \\ y &= \widetilde{C}(z) x \end{aligned} \quad (15)$$

by using $\widetilde{A}(z) = \sum_i^n \mu_i(z) \cdot A_i$, $\widetilde{B}(z) = \sum_i^n \mu_i(z) \cdot B_i$, etc.

3 OFFSET-ELLIMINATION PROCEDURE

Proposition 1 *Given a system with the structure in Definition* (1)*, for constant inputs $u = u^{eq}$, the equilibrium values of the state variables verify*

$$x_{ij}^{eq} = 0 \quad i = 1, \dots, p \;\; j = 2, \dots, r_i \quad (16)$$

Proof: With $u = u^{eq}$, the equilibrium equation is:

$$0 = Ax^{eq} + Bu^{eq} + R \quad (17)$$

Using the canonical matrix structure, the states $x_{i2} \dots x_{ir_i} \; \forall \; i$ verify:

$$0_{(r_i-1)\times 1} = I_{(r_i-1)\times(r_i-1)} \begin{bmatrix} x_{i2}^{eq} \\ \vdots \\ x_{ir_i}^{eq} \end{bmatrix} + 0_{(r_i-1)\times p} \begin{bmatrix} u_1^{eq} \\ \vdots \\ u_p^{eq} \end{bmatrix} + 0_{(r_i-1)\times 1} \quad (18)$$

Hence,

$$0 = I_{(r_i-1)\times(r_i-1)} \cdot \begin{bmatrix} x_{i2}^{eq} \\ \vdots \\ x_{ir_i}^{eq} \end{bmatrix} \quad i = 1, \dots, p \quad (19)$$

finally obtaining (16).

Proposition 2 *Given a system with the structure in Definition 1, with constant input $u = u^{eq}$, the equilibrium values for the state vector and the output are related, in the form:*

$$x^{eq} = [y_1^{eq} \; 0_{1\times(r_1-1)} \; y_2^{eq} \; 0_{1\times(r_2-1)} \; \cdots \; y_p^{eq} \; 0_{1\times(r_1-1)}]^T$$

$$y^{eq} = [y_1^{eq} \; \dots \; y_p^{eq}]^T$$

Proof: Replacing x^{eq} in the output equation,

$$y^{eq} = C \cdot x^{eq} \quad (20)$$

given the structure of C (7) and the results from the previous proposition, stating that only the states corresponding to the columns of the identity may be nonzero, we have:

$$y^{eq} = \begin{bmatrix} x_{11}^{eq} \\ x_{21}^{eq} \\ \vdots \\ x_{p1}^{eq} \end{bmatrix} \quad (21)$$

and finally,

$$x^{eq} = [y_1^{eq} \; 0_{1\times(r_1-1)} \; y_2^{eq} \; 0_{1\times(r_2-1)} \; \cdots \; y_p^{eq} \; 0_{1\times(r_p-1)}]^T$$

Lemma 1 *Given a canonical fuzzy system with the structure in Definition 2,* i.e.*,*

$$\begin{aligned} \dot{x} &= \widetilde{A}(z) \cdot x + \widetilde{B}(z) \cdot u + \widetilde{R}(z) \\ y &= \widetilde{C}(z) x \end{aligned} \quad (22)$$

defining an auxiliary input

$$u_{est}(z, y_{ref}) = (\widetilde{C}(z)\widetilde{A}(z)^{-1}\widetilde{B}(z))^{-1}(-y_{ref} - \widetilde{C}(z)\widetilde{A}(z)^{-1}\widetilde{R}(z)) \quad (23)$$

under suitable invertibility assumptions, and carrying out the change of variable

$$\hat{x} = x - x_{ref} \quad (24)$$

$$\hat{u} = u - u_{est}(z, y_{ref}) \quad (25)$$

where

$$x_{ref} = [y_{ref,1} \; 0_{1\times(r_1-1)} \; y_{ref,2} \; 0_{1\times(r_2-1)} \; \cdots \; \dots \; y_{ref,p} \; 0_{1\times(r_1-1)}]^T \quad (26)$$

and

$$y_{ref} = [y_{ref,1} \; \dots \; y_{ref,p}] \quad (27)$$

is a user-defined vector, then $\hat{x}=0$, $\hat{u}=0$ is the new equilibrium point for the system in the variables $\hat{x}$, $\hat{u}$ and, moreover, the transformed system has the equations:

$$\dot{\hat{x}} = \sum_i \mu_i(z)(A_i\hat{x} + B_i\hat{x}) \tag{28}$$

i.e., *it is a standard Takagi-Sugeno fuzzy system* (1), *where the offset terms have dissapeared. The variables $\hat{x}$, $\hat{u}$ will be denoted as* incremental.

Proof:

Let us denote by S_{z_0} the local affine system formed by the result of evaluating $\widetilde{A}$, $\widetilde{B}$, $\widetilde{C}$ and $\widetilde{R}$ at a particular (arbitrary) point z_0:

$$\begin{aligned} \dot{\xi} &= \widetilde{A}(z_0)\cdot\xi + \widetilde{B}(z_0)\cdot u + \widetilde{R}(z_0) \\ y &= \widetilde{C}(z_0)\xi \end{aligned} \tag{29}$$

which has the canonical structure of definition (1). Let us compute the input $u_{est} = u_{est}(z_0, y_{ref})$ so that output y_{ref} is an equilibrium point for the above system S_{z_0}, by using the equilibrium equation:

$$0 = \widetilde{A}(z_0)\cdot\xi_{ref} + \widetilde{B}(z_0)\cdot u_{est} + \widetilde{R}(z_0) \tag{30}$$

$$y_{ref} = \widetilde{C}(z_0)\cdot\xi_{ref} \tag{31}$$

On the following the dependence on z_0 will be omitted for notational simplicity. Carrying out some operations,

$$\begin{aligned} \widetilde{B}\cdot u_{est} &= -\widetilde{A}\cdot\xi_{ref} - \widetilde{R} \\ \widetilde{C}\widetilde{A}^{-1}\widetilde{B}\cdot u_{est} &= -\widetilde{C}\xi_{ref} - \widetilde{C}\widetilde{A}^{-1}\widetilde{R} = -y_{ref} - \widetilde{C}\widetilde{A}^{-1}\widetilde{R} \\ u_{est} &= (\widetilde{C}\widetilde{A}^{-1}\widetilde{B})^{-1}(-y_{ref} - \widetilde{C}\widetilde{A}^{-1}\widetilde{R}) \end{aligned} \tag{32}$$

Let's now obtain the equilibrium state ξ_{ref}. Indeed, as $u_{est}(z_0, y_{ref})$ ensures that the output y_{ref} is an equilibrium point, and as S_{z_0} has the structure 1, then Proposition 2 ensures that ξ_{ref} is:

$$\xi_{ref} = [y_{ref,1}\ 0_{1\times r_1}\ y_{ref,2}\ 0_{1\times r_2}\ \dots\ y_{ref,p}\ 0_{1\times r_1}]^T$$

identical to the state x_{ref} defined in (26). As z_0 in (30) is an arbitrary point, then

$$0 = \widetilde{A}(z)\cdot x_{ref} + \widetilde{B}(z)\cdot u_{est}(z, y_{ref}) + \widetilde{R}(z) \quad \forall\, z \tag{33}$$

Carrying out the change of variable

$$\hat{x} = x - x_{ref}$$

$$\hat{u} = u - u_{est}(z, y_{ref})$$

and using (33), the system equations may be written as

$$\begin{aligned} \dot{x} &= \widetilde{A}x + \widetilde{B}u + \widetilde{R} = \widetilde{A}x + \widetilde{B}\hat{u} + \widetilde{B}u_{est} + \widetilde{R} \\ \dot{x} &= \widetilde{A}x + \widetilde{B}\hat{u} - \widetilde{A}x_{ref} - \widetilde{R} + \widetilde{R} = \widetilde{A}(x - x_{ref}) + \widetilde{B}\hat{u} \\ \dot{x} &= \widetilde{A}\hat{x} + \widetilde{B}\hat{u} \end{aligned}$$

If y_{ref} is considered as a constant setpoint ($\dot{y}_{ref} = 0$), then $\dot{x}_{ref} = 0$ and, hence $\dot{\hat{x}} = \dot{x}$. So the fuzzy system in the new variables results in:

$$\dot{\hat{x}} = \widetilde{A}\hat{x} + \widetilde{B}\hat{u} = \quad \textstyle\sum_i \mu_i(z)(A_i\hat{x} + B_i\hat{x})$$

whose equilibrium point is $\hat{x} = 0$, corresponding to x_{ref} (and output y_{ref}) in the original non-incremental variables. □

The system in the new variables has its offset term removed and a standard fuzzy controller may be designed on it, such as the ones in (Tanaka and Wang, 2001) using LMI techniques, which will be used in the examples below. The control action for the original system will be computed by adding to the resulting control action the term $u_{est}(z, y_{ref})$.

Note also that, with an ordinary TS system and a setpoint $y = 0$, the result is $u_{est} = 0$, hence the proposed framework encompasses the standard one. For the canonical systems, it is more powerful, however, as setpoint changes can be immediately accommodated as the examples below will illustrate.

4 EXAMPLES

In this section, a set of examples showing the possibilities of the proposed approach will be presented. First, the control of a standard TS fuzzy system with no offset will be extended to varying operation points, in order to compare with the results applying usual methodologies involving a constant change of variable. Then, a second example will illustrate the proposed methodology in a MIMO case.

Example 1 *Let us have a standard, offset-free system defined by:*

$$\begin{aligned} \dot{x} &= \sum_{i=1}^{2} \mu_i(z)(A_i x + B_i u) \\ y &= Cx \end{aligned} \tag{34}$$

with the two models given by:

$$A_1 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 2 & 1 \end{bmatrix} \tag{35}$$

$$A_2 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 4 & 8 & 4 \end{bmatrix} \tag{36}$$

$$B_1 = B_2 = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T \tag{37}$$

$$C = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \tag{38}$$

and membership functions $\mu_i(z)$, defined on $z = x_1 + 2x_2 + x_3$ as the trapezoidal partition depicted in Figure 1. Figure 2 shows the nonlinearity in the system as a function of z. Note that conditions to design a standard PDC controller (Tanaka and Wang, 2001) for the TS system to reach the origin are fulfilled. For instance, an LMI methodology

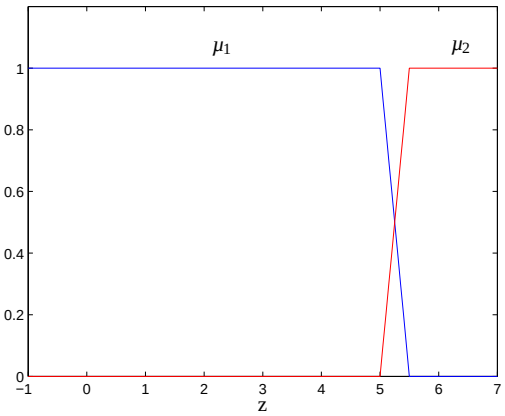


Figure 1: Membership functions.

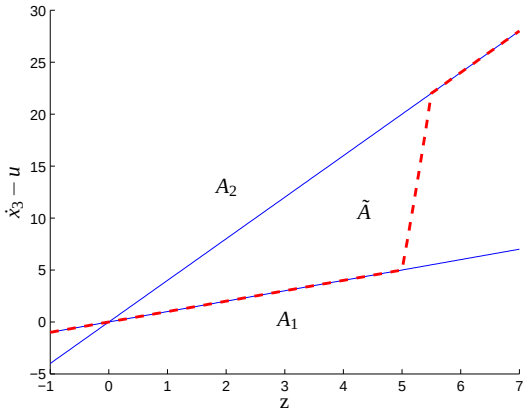


Figure 2: Nonlinearity in $\dot{x}_3 - u$.

(Tanaka and Wang, 2001) may be applied. To achieve a decay rate α, the following LMIs must be verified:

$$-XA_1^T - A_1X + M_1^T B_1^T + B_1M_1 - 2\alpha X > 0 \quad (39)$$
$$-XA_2^T - A_2X + M_2^T B_2^T + B_2M_2 - 2\alpha X > 0 \quad (40)$$
$$-XA_1^T - A_1X - XA_2^T - A_2X + M_1^T B_2^T + \quad (41)$$
$$+B_2M_1 + M_2^T B_1^T + B_1M_2 - 4\alpha X > 0 \quad (42)$$

where

$$X = P^{-1}, \;\; M_1 = F_1X, \;\; M_2 = F_2X \quad (43)$$

being P a quadratic matrix defining a Lyapunov function and F_1 and F_2 the state feedback gains to be implemented, *i.e.*, the control action:

$$\hat{u} = -(\mu_1(z)F_1 + \mu_2(z)F_2)\hat{x} \quad (44)$$

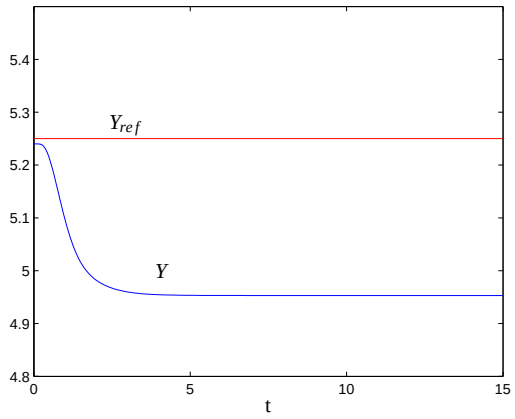


Figure 3: System output Y.

A set of LMI conditions (decay $\alpha = 1$) for the above system yields the controller:

$$F_1 = \begin{bmatrix} 27.5203 & 28.7108 & 8.4221 \end{bmatrix} \quad (45)$$
$$F_2 = \begin{bmatrix} 30.5203 & 34.7108 & 11.4221 \end{bmatrix} \quad (46)$$
$$\tilde{F}(z) = \sum_{i=1}^{2} \mu_i(z)F_i \quad (47)$$

which, as expected, behaves correctly when reaching the origin (figure not shown for brevity).

However, when trying to stabilise the system around a new operating point ($u_{ref} = -13.125$, $y_{ref} = 5.25$, $x_{ref} = (5.25, 0, 0)^T$), Lyapunov conditions no longer hold as the linearised model at that point has slopes out of those given by the vertices of the model above: in order to use a standard methodology in that case, redefining the local models would be needed. Indeed, with the constant change of variable $\hat{u} = u + 13.125$, $\hat{x} = x - x_{ref}$ (the usual one to achieve $x = 0$ as the operation point in many linear and fuzzy techniques), the resulting controller

$$u = -13.125 + \tilde{F}(z)(x - (5.25, 0, 0)^T) \quad (48)$$

yields an unstable equilibrium point: Figure 3 shows how initial conditions in the vicinity of the desired target drift away to another region of the state space.

On the contrary, the proposed methodology in this work provides a controller valid for all operating points with no modifications of the LMI conditions. Figure 4 shows the non-constant u_{est} calculated with lemma 1, which replaces the constant value -13.125 in (48) above. In that way, the resulting loop has the desired operating point as a stable equilibrium with the desired decay rate, as shown in Figure 5.

Example 2 *This example will demonstrate the methodology on a 5th order MIMO Takagi-Sugeno-Offset system with two unstable local models given by:*

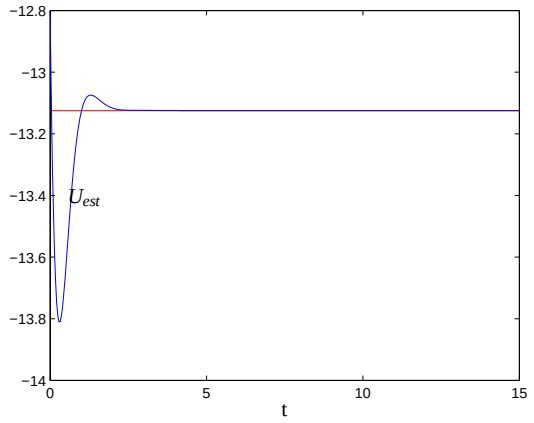


Figure 4: Non-constant u_{est} control action.

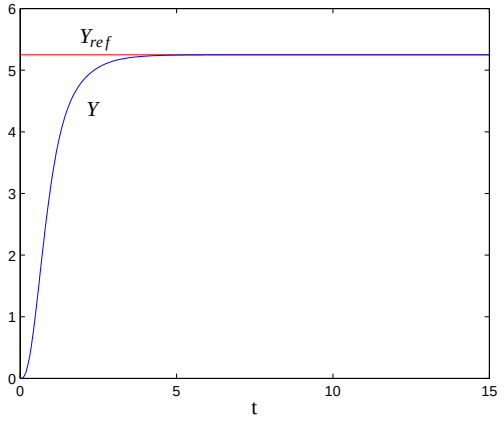


Figure 5: System output Y with non-constant u_{est}.

$$\begin{aligned} \dot{x} &= \sum_{i=1}^{2} \mu_i(z)(A_i(x-x_{i0})+B_i(u-u_{i0})) \\ y &= \sum_{i=1}^{2} \mu_i(z)C_i x \end{aligned} \tag{49}$$

where

$$A_1 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 4 & 2 & -5 & 3 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & -1 & -5 & -5 & -1 \end{bmatrix} \tag{50}$$

$$B_1 = \begin{bmatrix} 0 & 0 & 2 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}^T \tag{51}$$

$$A_2 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 5 & 1 & 0.1 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & -1 & -1 & -2 & -1 \end{bmatrix} \tag{52}$$

$$B_2 = \begin{bmatrix} 0 & 0 & 10 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}^T \tag{53}$$

$$C_1 = \begin{bmatrix} 1 & 0.1 & 0.05 & 0 & -0.1 \\ 0 & -0.2 & 0.5 & 1 & -0.1 \end{bmatrix} \tag{54}$$

$$C_2 = \begin{bmatrix} 1 & 0.2 & 0.1 & 0 & -0.2 \\ 0 & -0.1 & -0.5 & 1 & 0.2 \end{bmatrix} \tag{55}$$

$$x_{10} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T \tag{56}$$

$$u_{10} = \begin{bmatrix} 0 & 0 \end{bmatrix}^T \tag{57}$$

$$x_{20} = \begin{bmatrix} 2 & 0 & 0 & 2 & 0 \end{bmatrix}^T \tag{58}$$

$$u_{20} = \begin{bmatrix} 1 & 3 \end{bmatrix}^T \tag{59}$$

and the membership functions $\mu_i(z)$, defined on $z = x_1 + x_4$ as:

$$\mu_1(z) = \begin{cases} 1 & z < 0 \\ 1 - 0.25z & 0 \le z \le 4 \\ 0 & z > 4 \end{cases} \tag{60}$$

being $\mu_2(z) = 1 - \mu_1(z)$.

Let us group the different equilibrium points of each local model into an offset term:

$$R_i = -A_i x_{i0} - B_i u_{i0} \tag{61}$$

Hence, the system follows the structure in Definition 2. So the control action $u_{est}(z, y_{ref})$, computed via (23), and the equilibrium state are

$$x_{ref} = [y_{ref1}\ 0\ 0\ y_{ref2}\ 0]^T \tag{62}$$

$$u_{est} = (\widetilde{C}\widetilde{A}^{-1}\widetilde{B})^{-1}(-y_{ref} - \widetilde{C}\widetilde{A}^{-1}\widetilde{R}) \tag{63}$$

where y_{ref1} and y_{ref2} are arbitrary user-defined setpoints for the two plant outputs. As usual, the change of variable removes the offset terms so the system may be expressed as $\dot{\hat{x}} = \sum_{i=1}^{2} \mu_i(z)(A_i\hat{x} + B_i\hat{u})$. For a decay of $\alpha = 0.5$, the LMI Control Toolbox in Matlab obtains:

$$F_1 = \begin{bmatrix} 2.305 & -1.104 & -0.758 & 2.048 & 1.181 \\ 19.95 & 48.12 & 3.575 & -22.27 & -11.56 \end{bmatrix}$$

$$F_2 = \begin{bmatrix} 1.985 & -1.718 & -0.798 & 2.338 & 1.257 \\ -0.792 & 45.89 & 12.89 & -31.82 & -15.57 \end{bmatrix}$$

Hence the actual control action to be applied to the plant, after inverting the change of variable is:

$$u = u_{est}(z, y_{ref}) - (\mu_1(z)F_1 + \mu_2(z)F_2)(x - x_{ref}) \tag{64}$$

Figure 6 shows the system response approaching a setpoint $y_{ref} = [1\ 1]^T$. The usefulness of the obtained controller for setpoint changes is shown in Figure 10.

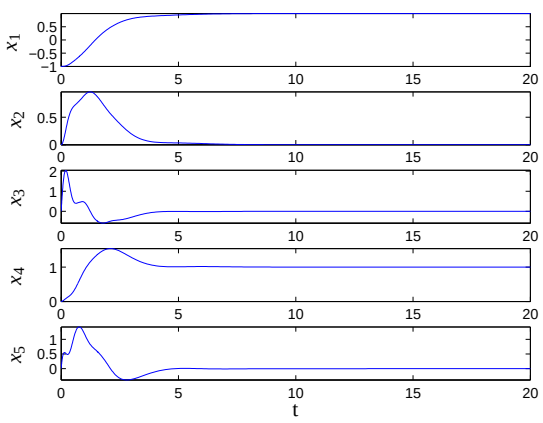


Figure 6: Time response of the state variables.

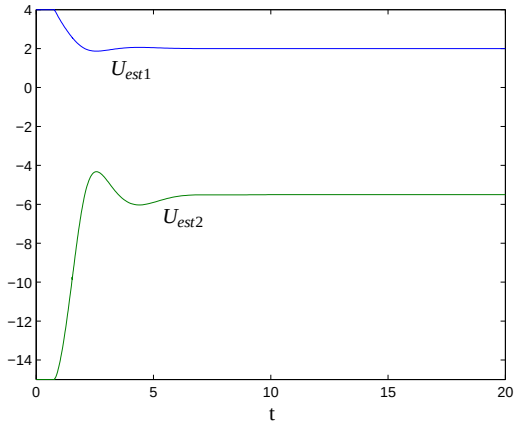


Figure 7: Offset removing term U_{est}.

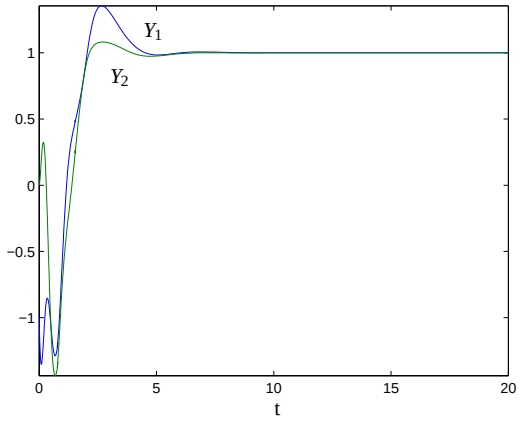


Figure 8: System output Y.

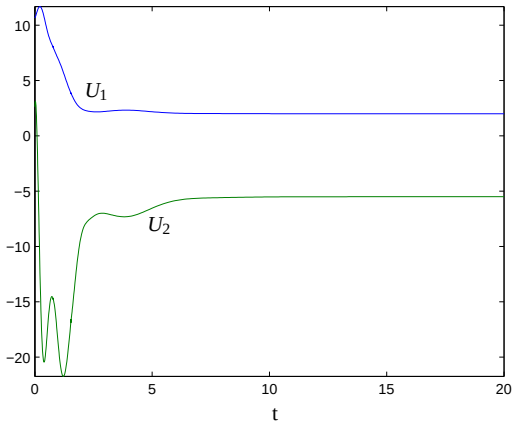


Figure 9: Actual overall control action U.

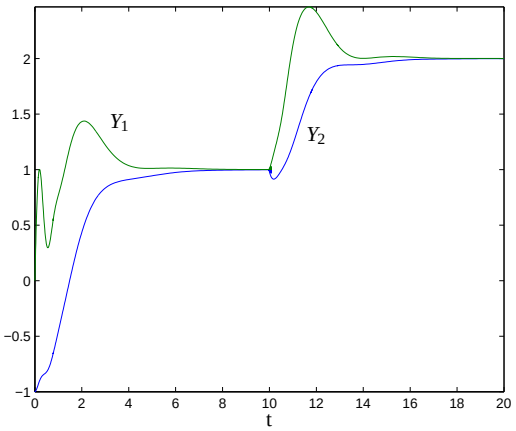


Figure 10: System output Y with a setpoint change to $Y_{ref} = (2,2)^T$.

5 CONCLUSIONS

This paper presents an offset-ellimination change of variable which applies to fuzzy Takagi-Sugeno-offset models with local linear models with a particular canonical structure. The canonical structure may be obtained, for instance, by taking as state variables the outputs and its derivatives.

As a result, a transformed system with equilibrium at $\hat{x} = 0$ is obtained. The difference with standard changes of variable is that it is non-constant in time. As a result, the offset is neatly removed and the resulting transformed system has the same representation for any desired setpoint, and well-known control design techniques for fuzzy non-offset Takagi-Sugeno systems may be directly applied independently of the

chosen setpoint. Interestingly, as a particular case, the procedure applies to setpoint changes in standard offset-free Takagi-Sugeno models.

The presented results apply to a state feedback setting. Further research should be devoted to generalising the procedure to situations with output feedback and noisy measurements.

REFERENCES

Antsaklis Panos, J. and Michel Anthony, N. (1997). *Linear Systems*. Ed. McGraw-Hill, New York, USA.

Babuska, R. (1998). *Fuzzy Modeling for Control*. Ed. Kluwer Academic, Boston, USA.

Boyd, S., ElGhaoui, L., Feron, E., and Balakrishnan, V. (1994). *Linear matrix inequalities in system and control theory*. Ed. SIAM, Philadelphia, USA.

Guerra, T. and Vermeiren, L. (2004). LMI-based relaxed nonquadratic stabilization conditions for nonlinear systems in the Takagi-Sugeno's form. *Automatica*, 40:823–829.

Johansson, M. (1999). Piecewise quadratic stability of fuzzy systems. *IEEE Trans. Fuzzy Syst.*, 7:713–722.

Kim, E. and Kim, S. (2002). Stabiliy analysis and synthesis for affine fuzzy control system via LMI and ILMI: Continous case. *IEEE Transactions on Fuzzy Systems*, 10:391–400.

Nelles, O., Fink, A., and Isermann, R. (2000). Local linear model trees (lolimot) toolbox for nonlinear system identification. In *Proc. 12th IFAC Symposium on System Identification*. Elsevier.

Sala, A., Guerra, T., and Babuska, R. (2005). Perspectives of fuzzy systems and control. *Fuzzy Sets and Systems*, page In Print.

Slotine, J.-J. E. and Li, W. (1991). *Applied Nonlinear Control*. Ed. Prentice Hall, Englewood Cliffs,New Jersey.

Takagi, T. and Sugeno, M. (1985). Fuzzy identification of systems and its applications to modeling and control. *IEEE Transactions on System, Man and Cybernetics*, 15:116–132.

Tanaka, K. and Wang, H. O. (2001). *Fuzzy control systems design and analysis*. Ed. John Wiley & Sons, New York, USA.

DISCRETE GENETIC ALGORITHM AND REAL ANT COLONY OPTIMIZATION FOR THE UNIT COMMITMENT PROBLEM

Guillaume Sandou
Supélec, Automatic Control Department – 3, rue Joliot Curie – 91192 Gif-sur-Yvette France
guillaume.sandou@supelec.fr

Keywords: Metaheuristics, unit commitment, ant colony, genetic algorithm, scheduling.

Abstract: In this paper, a cooperative metaheuristic for the solution of the Unit Commitment problem is presented. This problem is known to be a large scale, mixed integer problem. Due to combinatorial complexity, the exact solution is often intractable. Thus, a metaheuristic based method has to be used to compute a near optimal solution with low computation times. A new approach is presented here. The main idea is to couple a genetic algorithm to compute binary variables (on/off status of units), and an ant colony based algorithm to compute real variables (produced powers). Finally, results show that the cooperative method leads to the tractable computation of a satisfying solution for medium scale Unit Commitment problems.

1 INTRODUCTION

The Unit Commitment problem is a mixed integer problem, referring to the optimal scheduling of several production units, satisfying consumer's demand and technical constraints. Integer variables are the on/off status of production units, and real variables are produced powers. Numerous methods have been applied; see (Sen and Kothari, 1998).

The first idea is to use an exact solution method: exhaustive enumeration, "Branch and Bound" (Chen and Wang, 1993), dynamic programming (Ouyang and Shahidehpour, 1991). Due to temporal coupling of constraints (time up / time down constraints), a large temporal horizon is required, leading to a large number of binary variables: exact methods suffer from combinatorial complexity. Approximated methods are required for tractable results.

Deterministic approximated methods have been tested: priority lists in (Senjyu, et al., 2004) or expert systems. Due to numerous constraints, this kind of methods are often strongly suboptimal. Constraints are considered by the Lagrangian relaxation method, see (Zhai and Guan, 2002). Multi unit coupling constraints are relaxed. As a result, the unit Commitment problem is divided into several smaller optimization problems. However, due to the non convexity of the objective function, no guarantee can be given on the duality gap and the actual optimality of the solution. Further, an iterative procedure has to be used: solution of the optimization problems with fixed Lagrange multipliers, updates of these multipliers, and so on. This update can be performed with genetic algorithms as in (Cheng, et al., 2000) or by subgradient methods (Dotzauer, et al., 1999).

Stochastic approximated algorithms, called metaheuristics are potentially interesting methods for Unit Commitment as they are able to compute near optimal solutions with low computation times. A simulated annealing approach is used in (Yin Wa Wong, 1998), tabu search is used in (Rajan and Mohan, 2004) and genetic algorithms are used in (Kasarlis, *et al.*, 1996). Cooperative algorithms have been developed to couple the advantages of several optimization methods: genetic algorithms and simulated annealing are used in (Cheng, et al., 2002); simulated annealing and local search in (Purushothama and Jenkins, 2003).

In (Serban and Sandou, 2007), a mixed ant colony method has been proposed. The approach is interesting, but, due to the positive feedback of ant colony, may quickly converge to local minima. To circumvent this problem, a new cooperative strategy is defined in this paper. The idea is to use a knowledge based genetic algorithm for binary variables to achieve a deep exploration of the search space, and simultaneously an ant colony based algorithm for real variables.

The paper is organized as follows. In section 2, the Unit Commitment problem is briefly called up.

The cooperative metaheuristic ant colony/genetic algorithm method is depicted in section 3. Both algorithms are described, together with the definition of a criterion guaranteeing feasibility of the solution. Numerical results are given in section 4. Finally, concluding remarks are drawn in section 5.

2 UNIT COMMITMENT PROBLEM

The Unit Commitment problem is a classical large scale mixed integer optimization problem. Following notations are used throughout the paper:

- N: length of time horizon,
- n: (subscript) : time interval number n,
- K: number of production unit,
- k (superscript): production unit number k,
- u_n^k : on/off status of production unit k during time interval n (binary variable),
- Q_n^k : power produced by production unit k during time interval n (real variable).

2.1 Cost Function

The objective function is the sum of production, start-up, and shut-down costs for all time intervals and all units:

$$\min_{\{u_n^k, Q_n^k\}} \sum_{n=1}^{N} \left(\sum_{k=1}^{K} \begin{pmatrix} c_{prod}^k(Q_n^k, u_n^k) \\ + c_{on/off}^k(u_n^k, u_{n-1}^k) \end{pmatrix} \right), \quad (1)$$

where production cost of unit k can be expressed by:

$$c_{prod}^k(Q_n^k, u_n^k) = \alpha_2^k (Q_n^k)^2 + \alpha_1^k Q_n^k + \alpha_0^k u_n^k, \quad (2)$$

start-up cost and shut-down costs are:

$$c_{on/off}^k(u_n^k, u_{n-1}^k) = c_{on}^k u_n^k (1 - u_{n-1}^k) + c_{off}^k u_{n-1}^k (1 - u_n^k), \quad (3)$$

and α_2^k, α_1^k, α_0^k, c_{on}^k and c_{off}^k are technical data of production unit k.

2.2 Constraints

Constraints are:

- Capacity constraints

$$Q_{\min}^k u_n^k \le Q_n^k \le Q_{\max}^k u_n^k \quad (4)$$

- Consumers' demand satisfaction

$$\sum_{k=1}^{K} Q_n^k \ge Q_n^{dem} \quad (5)$$

- Time up and time down constraints

$$\begin{aligned} &(u_{n-1}^k = 0, u_n^k = 1) \\ &\quad \Rightarrow \left(u_{n+1}^k = 1, u_{n+2}^k = 1, \cdots, u_{n+T_{up}^k - 1}^k = 1 \right) \\ &(u_{n-1}^k = 1, u_n^k = 0) \\ &\quad \Rightarrow \left(u_{n+1}^k = 0, u_{n+2}^k = 0, \cdots, u_{n+T_{down}^k - 1}^k = 0 \right) \end{aligned} \quad (6)$$

Such constraints are temporally coupling constraints which express dynamics on production units. $Q_{\min}^k$, $Q_{\max}^k$, T_{up}^k and T_{down}^k are technical data.

3 COOPERATIVE METAHEURISTIC SOLUTION

3.1 Algorithm Principles

As already mentionned, Unit Commitment is a large scale mixed integer programming problem. Genetic algorithm is a well known algorithm for combinatorial optimization problems. In this study, a specific criterion is defined (see section 3.2), based on particular penalty functions to guarantee the solution feasibility. Genetic algorithm is used to compute binary variables and is depicted in section 3.3. Further, a stochastic algorithm is simultaneously used to compute real variables, based on ant colony optimization. It is presented in section 3.4.

3.2 Optimization Criterion

3.2.1 Criterion Expression

Consider that a feasible solution is known with a cost c^f. The following optimization criterion is defined:

$$\min_{\substack{y=(u_n^k, Q_n^k) \\ n=1,\cdots,N \\ k=1,\ldots,K}} \left(\sum_{n=1}^{N} \sum_{k=1}^{K} \begin{pmatrix} c_{prod}^k(Q_n^k, u_n^k) \\ + c_{on/off}^k(u_n^k, u_{n-1}^k) \end{pmatrix} + \begin{pmatrix} \\ ((1+\varepsilon)\, c^f + h(y)).B(y) \\ \\ \end{pmatrix} \right), \quad (7)$$

where:

- ε is a small positive real,
- $h(y)$ is a penalty function for non feasible solutions,

- $B(y)$ is a boolean function (1 for non feasible solutions and 0 for feasible ones).

With this criterion, any unfeasible solution has a higher cost than the feasible known solution: any unconstrained optimization algorithm can solve the problem. Thus, an elitist genetic algorithm can be used. The definition of criterion (7) only supposed that a feasible solution is known. It can be easily computed using a simple priority list. This is a very suboptimal solution, but the quality of this first feasible solution is not crucial, as the criterion can be updated when new feasible solutions are known.

3.2.2 Penalty Expression

The following variables are added:

$$\begin{aligned} \delta_n^k &= u_n^k (1 - u_{n-1}^k) \\ \varepsilon_n^k &= u_{n-1}^k (1 - u_n^k) \end{aligned} \tag{8}$$

With these variables, time-up and time-down constraints are expressed by linear expressions:

$$\begin{aligned} &\delta_n^k = 1 \Rightarrow (u_{n+1}^k = 1, \ldots, u^k_{n+T_{up}^k - 1} = 1) \\ &\Leftrightarrow \sum_{j=0}^{T_k^{up}-1} u_{n+j}^k \geq T_{up}^k \delta_n^k \end{aligned} \tag{9}$$

$$\begin{aligned} &\varepsilon_n^k = 1 \Rightarrow (u_{n+1}^k = 0, \ldots, u^k_{n+T_{down}^k - 1} = 0) \\ &\Leftrightarrow \sum_{j=0}^{T_k^{down}-1} (1 - u_{n+j}^k) \geq T_{down}^k \varepsilon_n^k \end{aligned} \tag{10}$$

Capacity constraints and consumers' demands satisfaction are linear. All constraints can be expressed by a linear equation, $A_c x \leq B_c$, where x is $(u_n^k, Q_n^k, \delta_n^k, \varepsilon_n^k; n = 1, \ldots, N; k = 1, \ldots, K)^T$, leading to a high tractability of the boolean and the penalty functions computation.

3.3 Genetic Algorithm for on/off Variables

3.3.1 Algorithm Principles

Genetic algorithm is a well known optimization method. Fig. 1 and 2 represent classical cross-over and mutation operators for Unit Commitment problem.

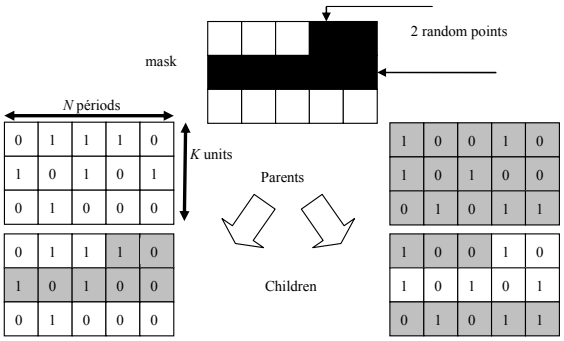


Figure 1: Classical crossing over operator.

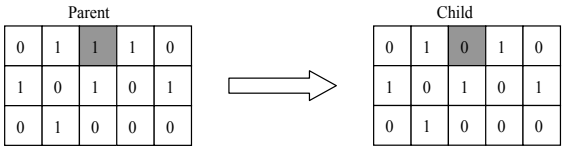


Figure 2: Classical mutation operator.

Individuals are in a matrix form. Crossover operator create 2 potentially low cost children from 2 parents by merging their variables (or genes). The mutation operator allows the introduction of new genes in the population by randomly changing one of the variables. Finally, the selection operator is performed with a classical biased roulette method.

3.3.2 Knowledge Based Operators

It has been observed that the genetic algorithm can be more efficient by using some knowledge based operators. New genetic operators are added, considering properties of the problem. The first operator is a "selective mutation operator". Consider unit scheduling of fig. 3. Due to time-up and time-down constraints, a classical mutation leads very often to an infeasible solution. To increase the probability of reaching a new feasible point, a "selective mutation operator" is introduced: this operator detects switching times and allows a random mutation only for these genes.

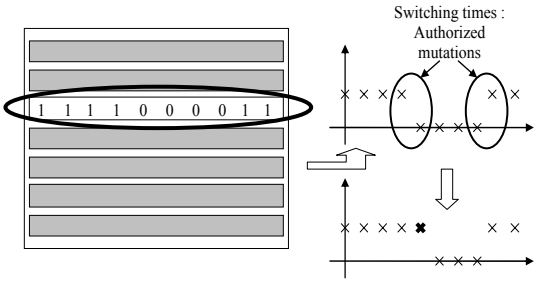


Figure 3: Selective mutation operator.

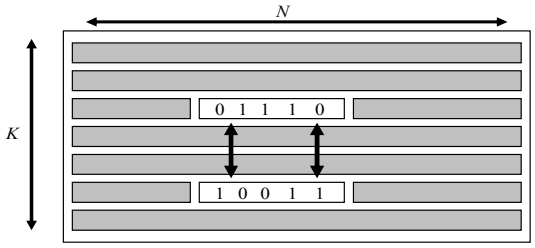


Figure 4: Exchange operator.

The second operator is an exchange operator, introduced by (Kasarlis, et al., 1996). Some production units are profitable or have larger capacities. It may be interesting to exchange a part of the planning of two production units (see fig. 4).

Finally, all-on and all-off operator are introduced. Consider fig. 5. If the unit has a time down constraints of two hours, it may be difficult to go from feasible point a) to feasible point b). The all-on (resp. all-off) operator randomly select two time intervals and a production unit and switch on (resp. switch off) the production unit between these time intervals: the idea is to increase the probability of "crossing the infeasible space".

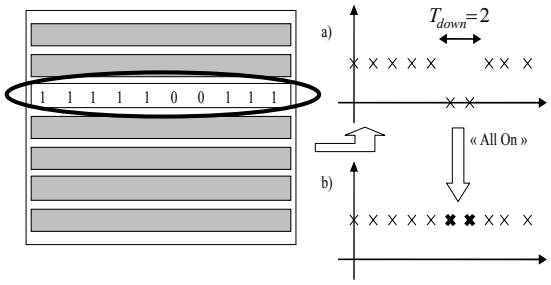


Figure 5: All-on operator.

3.4 Continuous Ant Colony Optimization for Produced Powers

Ant colony optimization was firstly introduced by Marco Dorigo. Ants' behaviour has been used as a metaphor to design algorithms for combinatorial optimization problems such as the Travelling Salesman Problem (Dorigo, et al. 1997). Extensions for continuous search spaces have been proposed by (Socha and Dorigo, 2006) and have been used in (Serban and Sandou, 2007) in a pure ant colony solution of Unit Commitment. Results are here called up. For each binary solution, $U = (u_n^k; n = 1,\ldots,N; k = 1,\ldots,K)$, real values $Q = (Q_n^k; n = 1,\ldots,N; k = 1,\ldots,K) = (x^1,\ldots,x^{KN})$ have to be associated. To compute these real variables, a matrix **T** of s real solutions, called "archive matrix of solutions", is stored:

$$\mathbf{T} = \begin{bmatrix} x_1^1 & x_1^2 & \ldots & x_1^i & \cdots & x_1^{KN} \\ x_2^1 & x_2^2 & \ldots & x_2^i & \cdots & x_2^{KN} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ x_j^1 & x_j^2 & \cdots & x_j^i & \cdots & x_j^{KN} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ x_s^1 & x_s^2 & \ldots & x_s^i & \ldots & x_s^{KN} \end{bmatrix}. \quad (11)$$

These solutions are evaluated with respect to the objective function (1). Costs are stored in **H**:

$$\mathbf{H} = \begin{bmatrix} f_1 & f_2 & \ldots & f_j & \ldots & f_s \end{bmatrix}^T, \text{ with}$$

$$f_j = f(U_j, Q_j) = \sum_{n=1}^{N}\left(\sum_{k=1}^{K}\begin{pmatrix} c_{prod}^k(Q_n^k, u_n^k) + \\ c_{on/off}^k(u_n^k, u_{n-1}^k) \end{pmatrix}\right). \quad (12)$$

The solutions are sorted according to their costs. From these costs, weights are defined according to the ranks of the solutions in the matrix. For the solution with rank r, the weight is defined by:

$$\omega_r = \frac{1}{qs\sqrt{2\pi}} e^{-\frac{(r-1)^2}{2q^2s^2}}. \quad (13)$$

Finally, a discrete probability distribution is defined from these weights:

$$p_r = \omega_r / \sum_{j=1}^{s} \omega_j \quad (14)$$

q is a tuning parameter of the algorithm. To compute a new real solution, the following procedure is performed:

- a "model ant", say l, is chosen, according to this discrete probability distribution (14).
- Each real variable x_{new}^i, i = 1,…,KN, is chosen with a Gaussian probability whose mean and standard deviation is computed by:

$$\begin{cases} \mu_{new}^i = x_l^i \\ \sigma_{new}^i = \dfrac{\xi}{s-1}\displaystyle\sum_{m=1}^{s}\left|x_m^i - x_l^i\right| \end{cases}. \quad (15)$$

ξ is also a tuning parameter of the algorithm. When real variables have been chosen, consumer's demands (5) may not be fulfilled. Furthermore, the selection of produced powers may lead to overproduction. To get rid of these problems, the following improvement procedure is used:

- Select Q_n^k with the previous algorithm.
- If $\sum_{k=1}^{K} Q_n^k u_n^k > Q_n^{dem}$ (resp. $<$), then randomly choose, if possible, one of switched on units, and decrease (resp. increase) the corresponding produced power until

$\sum_{k=1}^{K} Q_n^k u_n^k = Q_n^{dem}$. If it is not sufficient, choose several production units, if possible.

When all new solutions have been computed, the best new solutions are stored in matrix **T**, replacing solutions whose costs were too high. This is an analogy with physical evaporation of pheromone.

4 NUMERICAL RESULTS

4.1 Algorithm Implementation

The proposed cooperative method has been tested with Matlab 6.5 with a Pentium IV 2.5 GHz. When the stochastic cooperative algorithm is completed, a final local search is performed: binary values are set to their final values, and a real optimization based on Semi Definite Programming is performed to solve this particular economic dispatch problem. As stochastic algorithms are considered, 70 tests are performed, and statistical data about the results are given. Optimization horizon is 24 hours with a sampling time of one hour.

4.2 "4 unit" Academic Example

A 4 unit case is considered (see table 1).

Table 1: Characteristics for the "4 unit case".

	$Q_{\min}$ (MW)	$Q_{\max}$ (MW)	α_0 (€)	α_1	c_{on} (€)	c_{off} (€)	T_{down} (h)	T_{up} (h)
1	10	40	25	2.6	10	2	2	4
2	10	40	25	7.9	10	2	2	4
3	10	40	25	13.1	10	2	3	3
4	10	40	25	18.3	10	2	3	3

At time 0, all units are switched off and can be switched on. Note that linear costs have been chosen (α_2 = 0). For this relative small scale cases, and for linear costs, an exact solution has been computed by "Branch and Bound". Consumer's demand is depicted in fig. 6. This demand can be fulfilled by 2 production units (see 2 units limit in fig. 6), except for hour number 9. Because of time up constraints this unit will be switched on for 3 hours. The optimal solution is given in fig. 7.

The corresponding optimization problem is made of 96 binary variables (24 hours and 4 units) and 96 real variables. Table 2 shows results of optimization. Statistical results are given: best case, mean, number of success (a test is successful if the best solution is found).

The following parameters were used:

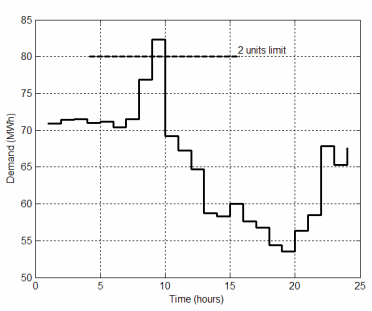


Figure 6: Consumer's demand.

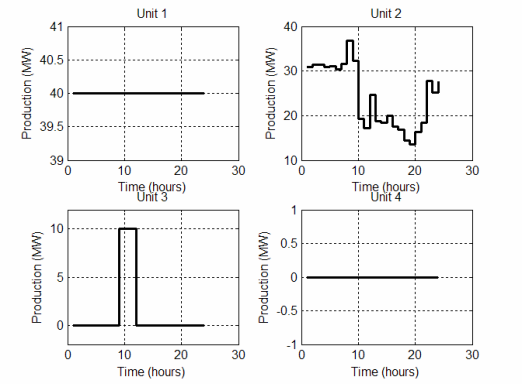


Figure 7: Optimal solution for "4 unit case".

- Genetic population size: 50,
- Cross-over rate: 70%,
- Mutation rate: 5%,
- Knowledge based operators rate: 10%,
- Archive matrix size: $s = 20$,
- Tuning parameters $q = 1; \xi = 0.8$.

Results show that interesesting solutions can be computed with relatively low computation times.

Table 2: Optimization results "4 unit case".

Case	Best	Mean	Nb. Success	Time
100 iter.	8778 € (+0%)	9449 € (+7.6%)	8/70	22 s
200 iter.	8778 € (+0%)	9004 € (+2.6%)	32/70	45 s
500 iter.	8778 € (+0%)	8922 € (+1.6%)	45/70	115 s

4.3 Medium Scale Case

A "10 unit" case is now considered (see table 3). This a medium scale case. Low start up and start down costs have been considered, leading to the possibility of guessing the optimal solution. The corresponding optimal cost is 29795 €. For a 24 hour optimization, this problem is made of 240 binary

optimization variables and 240 real variables. Results for the cooperative method are given in table 4. As in previous examples, 70 tests are performed and statistical results are given (best case, mean). The same values were used for parameters.

Table 3: Characteristics for the "10 unit case".

	Q_{min} MW	Q_{max} MW	α_0 €	α_1	c_{on} €	c_{off} €	T_{down} h	T_{up} h
1	10	40	25	2.6	10	2	2	4
2	10	40	25	5.2	10	2	2	4
3	10	40	25	7.9	10	2	3	6
4	10	40	25	10.5	10	2	3	6
5	10	40	25	13.1	10	2	3	4
6	10	40	25	15.7	10	2	3	4
7	10	40	25	18.3	10	2	3	4
8	10	40	25	21.0	10	2	3	4
9	10	40	25	23.6	10	2	3	4
10	10	40	25	26.2	10	2	3	4

Results show the viability of the cooperative method to solve mixed integer optimization problems. Low computation times are observed, even for this medium scale case.

Table 4: Optimization results "10 unit case".

	Best	**Mean**	**Time**
500 iter.	30210 € (+1.4%)	32695 € (+9.7%)	275 s
1000 iter.	29851 € (+0.2%)	32138 € (+7.8%)	550 s

5 CONCLUSION

In this paper, a cooperative method ant colony/genetic algorithm for Unit Commitment solution has been proposed. The main idea is to use a genetic algorithm with knowledge based operators to compute binary variables and a real ant colony algorithm to compute real variables. To guarantee the feasibility of the final solution, a criterion has also been defined. Finally, the proposed method leads to near optimal solutions, with guarantees of feasibility and with low computation times.

Some dedicated methods are able to find better solutions than the proposed cooperative algorithm, and can consider larger scale cases. However, this cooperative method seems to be promising and the study has proven its viability.

Forthcoming works deal with the use of such a cooperative metaheuristic method to solve generic non linear mixed integer optimization problems, as the use of the method does not require any structural property of the problem.

REFERENCES

Chen C.-L and Wang S.-C. (1993), Branch and Bound scheduling for thermal generating units, *IEEE Trans. on Energy Conversion*, Vol. 8(2), pp.184-189.

Cheng C.-P., Liu C.-W., Liu C.-C. (2000), Unit Commitment by Lagrangian Relaxation and Genetic Algorithms, *IEEE Trans. on Power Systems*, Vol. 15(2), pp. 707-714.

Cheng C.-P., Liu C.-W., Liu C.-C. (2002), Unit Commitment by annealing-genetic algorithm, *Electrical Power and Energy Systems*, Vol. 24, pp. 149-158.

Dorigo M., Gambardella, L. M. (1997), Ant Colony System: a Cooperative Learning Approach to the Traveling Salesman Problem, *IEEE Trans. on Evolutionary Computation*, Vol. 1, pp. 53-66.

Dotzauer E., Holmström K., Ravn H. F. (1999), Optimal Unit Commitment and Economic Dispatch of Cogeneration Systems with a Storage, *Proceedings of the 13th Power Systems Computation Conference 1999 PSCC'99*, Trondheim, Norway, pp. 738-744.

Kasarlis S. A., Bakirtzis A. G. and Petridis V. (1996), A genetic algorithm solution to the unit commitment problem, *IEEE Trans. on Power Systems*, Vol. 11(1), pp. 83-92.

Ouyang Z. and Shahidehpour S. M. (1991), An intelligent dynamic programming for unit commitment application, *IEEE Trans. on Power Systems*, Vol. 6(3), pp. 1203-1209.

Purushothama G. K., Jenkins L. (2003), Simulated annealing with local search – a hybrid algorithm for Unit Commitment, *IEEE Trans. on Power Systems*, Vol. 18(1), pp. 273-278.

Rajan C. C. A and Mohan M. R. (2004), An evolutionary programming-based tabu search method for solving the unit commitment problem, *IEEE Trans. on Power Systems*, Vol. 19(1), pp. 577-585.

Sen S., Kothari D. P. (1998), Optimal Thermal Generating Unit Commitment: a Review, *Electrical Power & Energy Systems*, Vol. 20(7), pp. 443-451.

Senjyu T., Shimabukuro, K., Uezato K. and Funabashi T. (2004), A fast technique for Unit Commitment problem by extended priority list, *IEEE Trans. on Power Systems*, Vol. 19(4), pp. 2119-2120.

Serban A. T, Sandou G. (2007), Mixed ant colony optimisation for the Unit Commitment problem, *Lecture Notes in Computer Science*, n°4431/4432, pp. 332-340.

Socha K., Dorigo M. (2006), Ant colony optimization for continuous domains, Accepted to special issue of EJOR on adapting metaheuristics to continuous optimization.

Yin Wa Wong S. (1998), An Enhanced Simulated Annealing Approach to Unit Commitment, *Electrical Power & Energy Systems*, Vol. 20(5), pp. 359-368.

Zhai Q; and Guan X. (2002), Unit Commitment with identical units: successive subproblems solving method based on Lagrangian relaxation, *IEEE Trans. on Power Systems*, Vol. 17(4), pp. 1250-1257.

SENSOR-ASSISTED ADAPTIVE MOTOR CONTROL UNDER CONTINUOUSLY VARYING CONTEXT

Heiko Hoffmann, Georgios Petkos, Sebastian Bitzer and Sethu Vijayakumar
Institute of Perception, Action and Behavior, School of Informatics, University of Edinburgh, Edinburgh, UK
heiko@clmc.usc.edu, g.petkos@sms.ed.ac.uk, s.bitzer@ed.ac.uk, sethu.vijayakumar@ed.ac.uk

Keywords: Adaptive control, context switching, Kalman filter, force sensor, robot simulation.

Abstract: Adaptive motor control under continuously varying context, like the inertia parameters of a manipulated object, is an active research area that lacks a satisfactory solution. Here, we present and compare three novel strategies for learning control under varying context and show how adding tactile sensors may ease this task. The first strategy uses only dynamics information to infer the unknown inertia parameters. It is based on a probabilistic generative model of the control torques, which are linear in the inertia parameters. We demonstrate this inference in the special case of a single continuous context variable – the mass of the manipulated object. In the second strategy, instead of torques, we use tactile forces to infer the mass in a similar way. Finally, the third strategy omits this inference – which may be infeasible if the latent space is multi-dimensional – and directly maps the state, state transitions, and tactile forces onto the control torques. The additional tactile input implicitly contains all control-torque relevant properties of the manipulated object. In simulation, we demonstrate that this direct mapping can provide accurate control torques under multiple varying context variables.

1 INTRODUCTION

In feed-forward control of a robot, an internal inverse-model of the robot dynamics is used to generate the joint torques to produce a desired movement. Such a model always depends on the context in which the robot is embedded, its environment and the objects it interacts with. Some of this context may be hidden to the robot, e.g., properties of a manipulated object, or external forces applied by other agents or humans.

An internal model that does not incorporate all relevant context variables needs to be relearned to adapt to a changing context. This adaptation may be too slow since sufficiently many data points need to be collected to update the learning parameters. An alternative is to learn different models for different contexts and to switch between them (Narendra and Balakrishnan, 1997; Narendra and Xiang, 2000; Petkos et al., 2006) or combine their outputs according to a predicted error measure (Haruno et al., 2001; Wolpert and Kawato, 1998). However, the former can handle only previously-experienced discrete contexts, and the latter have been tested only with linear models.

The above studies learn robot control based purely on the dynamics of the robot; here, we demonstrate the benefit of including tactile forces as additional input for learning *non-linear* inverse models under *continuously-varying* context. Haruno et al. (Haruno et al., 2001) already use sensory information, but only for mapping a visual input onto discrete context states.

Using a physics-based robot-arm simulation (Fig. 1), we present and compare three strategies for learning inverse models under varying context. The first two infer the unknown property (hidden context) of an object during its manipulation (moving along a given trajectory) and immediately use this estimated property for computing control torques.

In the first strategy, only dynamic data are used, namely robot state, joint acceleration, and joint torques. The unknown inertia parameters of an object are inferred using a probabilistic generative model of the joint torques.

In the second strategy, we use instead of the

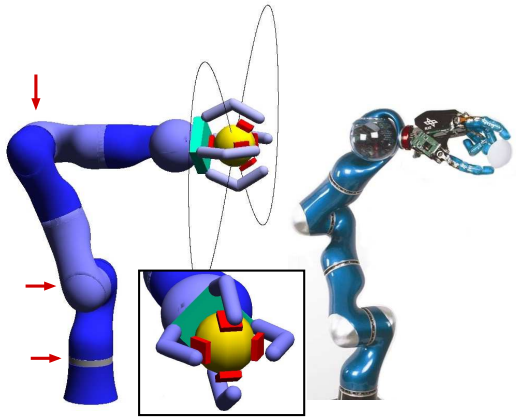


Figure 1: Simulated robot arm with gripper and force sensors and its real counterpart, the DLR light-weight arm III. Arrows indicate the three active joints used for the experiments. The curve illustrates the desired trajectory of the ball.

torques the tactile forces exerted by the manipulated object. The same inference as above can be carried out given that the tactile forces are linear in the object's mass. We demonstrate both of these steps with mass as the varying context. If more context variables are changing, estimating only the mass as hidden context is insufficient. Particularly, if also the mass distribution changes, both the center of mass and inertia tensor vary, leading to a high-dimensional latent variable space, in which inference may be infeasible given a limited number of data points.

In our third strategy, we use a direct mapping from robot state, joint accelerations, and tactile forces onto control torques. This mapping allows accurate control even with more than one changing context variable and without the need to extract these variables explicitly.

The remainder of this article is organized as follows. Section 2 briefly introduces the learning of inverse models under varying context. Section 3 describes the inference of inertia parameters based only on dynamic data. Section 4 describes the inference of mass using tactile forces. Section 5 motivates the direct mapping. Section 6 describes the methods for the simulation experiments. Section 7 shows the results of these experiments; and Section 8 concludes the article.

2 LEARNING DYNAMICS

Complex robot structure and non-rigid body dynamics (e.g, due to hydraulic actuation) can make analytic solutions to the dynamics inaccurate and cumbersome to derive. Thus, in our approach, we learn the dynamics for control from movement data. Particularly, we learn an inverse model, which maps the robot's state (here, joint angles θ and their velocities $\dot{\theta}$) and its desired change (joint accelerations $\ddot{\theta}$) onto the motor commands (joint torques τ) that are required to produce this change,

$$\tau = \mu(\theta, \dot{\theta}, \ddot{\theta}) \quad . \tag{1}$$

Under varying context, learning (1) is insufficient. Here, the inverse model depends on a context variable π,

$$\tau = \mu(\theta, \dot{\theta}, \ddot{\theta}, \pi) \quad , \tag{2}$$

In Sections 3 and 4, we first infer the hidden context variable and then plug this variable into function (2) to compute the control torques. In Section 5, the context variable π is replaced by sensory input that implicitly contains the hidden context.

3 INFERRING CONTEXT FROM DYNAMICS

During robot control, hidden inertia parameters can be inferred by observing control torques and corresponding accelerations (Petkos and Vijayakumar, 2007). This inference can be carried out efficiently because of a linear relationship in the dynamics, as shown in the following.

3.1 Linearity in Robot Dynamics

The control torques τ of a manipulator are linear in the inertia parameters of its links (Sciavicco and Siciliano, 2000). Thus, μ can be decomposed into

$$\tau = \Phi(\theta, \dot{\theta}, \ddot{\theta})\,\pi \quad . \tag{3}$$

Here, π contains the inertia parameters, $\pi = [m_1, m_1 l_{1x}, m_1 l_{1y}, m_1 l_{1z}, J_{1xx}, J_{1xy}, \ldots, m_n, m_n l_{nx}, m_n l_{ny}, m_n l_{nz}, J_{nxx}, \ldots, J_{nzz}]$, where m_i is the mass of link i, l_i its center-of-mass, and J_i its inertia tensor. The dynamics of a robot holding different objects only differs in the π parameters of the combination of object and end-effector link (the robot's link in contact with the object). To obtain Φ, we need to know for a set of contexts c the inertia parameters π_c and the corresponding torques τ_c. Given a sufficient number of τ_c and π_c values, we can compute Φ using ordinary least squares. After computing Φ, the robot's dynamics can be adjusted to different contexts by multiplying Φ with the inertia parameters π. The

following two sections show how to estimate π once Φ has been found.

3.2 Inference of Inertia Parameters

Given the linear equation (3) and the knowledge of Φ, the inertia parameters π can be inferred. Assuming Gaussian noise in the torques τ, the probability density $p(\tau|\theta,\dot{\theta},\ddot{\theta},\pi)$ equals

$$p(\tau|\theta,\dot{\theta},\ddot{\theta},\pi) = \mathcal{N}(\Phi\pi,\Sigma) \quad , \tag{4}$$

where $\mathcal{N}$ is a Gaussian function with mean $\Phi\pi$ and covariance Σ. Given $p(\tau|\theta,\dot{\theta},\ddot{\theta},\pi)$, the most probable inertia parameters π can be inferred using Bayes' rule, e.g., by assuming a constant prior probability $p(\pi)$:

$$p(\pi|\tau,\theta,\dot{\theta},\ddot{\theta}) \propto p(\tau|\theta,\dot{\theta},\ddot{\theta},\pi) \quad . \tag{5}$$

3.3 Temporal Correlation of Inertia Parameters

The above inference ignores the temporal correlation of the inertia parameters. Usually, however, context changes infrequently or only slightly. Thus, we may describe the context π at time step $t+1$ as a small random variation of the context at the previous time step:

$$\pi_{t+1} = \pi_t + \varepsilon_{t+1} \quad , \tag{6}$$

where ε is a Gaussian-distributed random variable with constant covariance Ω and zero mean. Thus, the transition probability $p(\pi_{t+1}|\pi_t)$ is given as

$$p(\pi_{t+1}|\pi_t) = \mathcal{N}(\pi_t,\Omega) \quad . \tag{7}$$

Given the two conditional probabilities (4) and (7), the hidden variable π can be described with a Markov process (Fig. 2), and the current estimate π_t can be updated using a Kalman filter (Kalman, 1960). Written with probability distributions, the filter update is

$$p(\pi_{t+1}|\tau^{t+1},x^{t+1}) = \eta\, p(\tau_{t+1}|x_{t+1},\pi_{t+1}) \cdot \int p(\pi_{t+1}|\pi_t)\, p(\pi_t|\tau^t,x^t)\, d\pi_t \quad , \tag{8}$$

where η is a normalization constant, and x stands for $\{\theta,\dot{\theta},\ddot{\theta}\}$ to keep the equation compact. A variable with superscript t stands for all observations (of that variable) up to time step t.

3.4 Special Case: Inference of Mass

We will demonstrate the above inference in the case of object mass m as the hidden context. This restriction

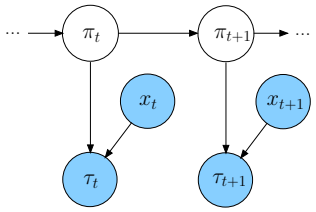


Figure 2: Hidden Markov model for dependence of torques τ on context π. Here, the state and state transitions are combined to a vector $x = \{\theta,\dot{\theta},\ddot{\theta}\}$.

essentially assumes that the shape and center of mass of the manipulated object are invariant. If m is the only variable context[1], the dynamic equation is linear in m,

$$\tau = g(\theta,\dot{\theta},\ddot{\theta}) + m\,h(\theta,\dot{\theta},\ddot{\theta}) \quad . \tag{9}$$

In our experiments, we first learn two mappings $\tau_1(\theta,\dot{\theta},\ddot{\theta})$ and $\tau_2(\theta,\dot{\theta},\ddot{\theta})$ for two given contexts m_1 and m_2. Given these mappings, g and h can be computed.

To estimate m, we plug (4) and (7) into the filter equation (8) and use (9) instead of (3). Furthermore, we assume that the probability distribution of the mass at time t is a Gaussian with mean m_t and variance Q_t. The resulting update equations for m_t and Q_t are

$$m_{t+1} = \frac{h^T\Sigma^{-1}(\tau-g) + \frac{m_t}{Q_t+\Omega}}{\frac{1}{Q_t+\Omega} + h^T\Sigma^{-1}h} \quad , \tag{10}$$

$$Q_{t+1} = \left(\frac{1}{Q_t+\Omega} + h^T\Sigma^{-1}h\right)^{-1} \quad . \tag{11}$$

Section 7 demonstrates the result for this inference of m during motor control. For feed-forward control, we plug the inferred m into (9) to compute the joint torques τ.

4 INFERRING CONTEXT FROM TACTILE SENSORS

For inferring context, tactile forces measured at the interface between hand and object may serve as a substitute for the control torques. We demonstrate this inference for the special case of object mass as context.

[1]This assumption is not exactly true in our case. A changing object mass also changes the center of mass and inertia tensor of the combination end-effector link plus object. Here, to keep the demonstration simple, we make a linear approximation and ignore terms of higher order in m – the maximum value of m was about one third of the mass of the end-effector link.

The sensory values s_i (the tactile forces) are linear in the mass m held in the robot's hand, as shown in the following. In the reference frame of the hand, the acceleration a of an object leads to a small displacement dx (Fig. 3).

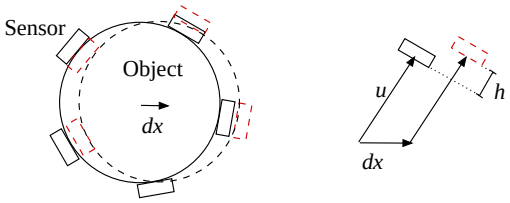


Figure 3: A force on the object held in the robot's hand leads to a displacement dx. This displacement shifts each sensor at position u (relative to the object's center) by h.

This displacement pushes each sensor by the amount h_i depending on the sensor's position u_i. Let e_i be a vector of unit length pointing in the direction of u_i, then $h_i = e_i^T dx$. Our sensors act like Hookean springs; thus, the resulting force equals $f_i = \kappa h_i e_i$, with κ being the spring constant. Since the object is held such that it cannot escape the grip, the sum of sensory forces f_i must equal the inertial force ma,

$$ma = \sum_{i=1}^{n} f_i = \kappa \sum_i (e_i^T dx)\, e_i \quad . \tag{12}$$

This linear equation allows the computation of dx,

$$dx = \frac{m}{\kappa}(E^T E)^{-1} a \quad , \tag{13}$$

where E is a matrix with row vectors e_i. Thus, each f_i is proportional to m. The total force measured at a sensor equals f_i plus a constant grip force (whose sum over all sensors equals zero). Therefore, the sensory values s can be written as

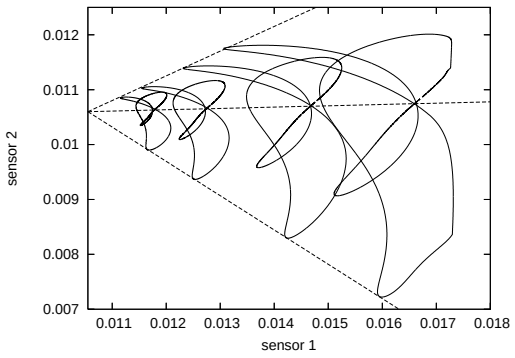


Figure 4: Two-dimensional projection of sensor values during figure-8 movements with four different masses. From left to right, the mass increases as 0.005, 0.01, 0.02, and 0.03.

$$s = s_0 + m\varphi(\theta, \dot{\theta}, \ddot{\theta}) \quad , \tag{14}$$

where φ is a function depending on the state and acceleration of the robot arm. This linearity is illustrated in Fig. 4 using data from our simulated sensors. Based on (14), the same inference as in Section 3 is possible using (10) and (11), and the estimated m can be used for control by using (9).

5 DIRECT MAPPING

The above inference of mass fails if we have additional hidden context variables, e.g., if the robot hand holds a dumb-bell, which can swing. In this general case, we could still use the inference based on dynamics. However, since for the combination of end-effector and object, both the inertia tensor and the center of mass vary, we need to estimate 10 hidden context variables (Sciavicco and Siciliano, 2000). Given the limited amount of training data that we use in our experiments, we expect that inference fails in such a high-dimensional latent space.

As alternative, we suggest using the sensory values as additional input for learning feed-forward torques. Thus, the robot's state and desired acceleration are augmented by the sensory values, and this augmented input is directly mapped on the control torques. This step avoids inferring unknown hidden variables, which are not our primary interest; our main goal is computing control torques under varying context.

The sum of forces measured at the tactile sensors equals the force that the manipulated object exerts on the robotic hand. This force combined with the robot's state and desired acceleration is sufficient information for predicting the control torques. Thus, tactile forces contain the relevant effects of an unknown varying context. This context may be, e.g., a change in mass, a change in mass distribution, or a human exerting a force on the object held by the robot.

Depending on the number of sensors, the sensory input may be high-dimensional. However, since the sensors encode only a force vector, the intrinsic dimensionality is only three. For learning the mapping $\tau(\theta, \dot{\theta}, \ddot{\theta}, s)$, regression techniques exist, like locally-weighted projection regression (Vijayakumar et al., 2005), that can exploit this reduced intrinsic dimensionality efficiently. In the following, we demonstrate the validity of our arguments in a robot-arm simulation.

6 ROBOT SIMULATION

This section describes the methods: the simulated robot, the simulated tactile sensors, the control tasks, the control architecture, and the learning of the feed-forward controller.

6.1 Simulated Robot Arm

Our simulated robot arm replicates the real Light-Weight Robot III designed by the German Aerospace Center, DLR (Fig. 1). The DLR arm has seven degrees-of-freedom; however, only three of them were controlled in the present study; the remaining joints were stiff. As end-effector, we attached a simple gripper with four stiff fingers; its only purpose was to hold a spherical object tightly with the help of five simulated force sensors. The physics was computed with the Open Dynamics Engine (`http://www.ode.org`).

6.2 Simulated Force Sensors

Our force sensors are small boxes attached to damped springs (Fig. 1). In the simulation, damped springs were realized using slider joints, whose positions were adjusted by a proportional-derivative controller. The resting position of each spring was set such that it was always under pressure. As sensor reading s, we used the current position of a box (relative to the resting position).

6.3 Control Tasks

We used two tasks: moving a ball around an eight figure and swinging a dumb-bell. In the first, one context variable was hidden, the mass of the ball. The maximum mass ($m = 0.03$) of a ball was about one seventh of the total mass of the robot arm. In the second task, two variables were hidden: the dumb-bell mass and its orientation. The two ball masses of the dumb-bell were equal.

In all tasks, the desired trajectory of the end-effector was pre-defined (Figs. 1 and 5) and its inverse in joint angles pre-computed. The eight was traversed only once lasting 5000 time steps, and for the dumb-bell, the end-effector swung for two periods together lasting 5000 time steps and followed by 1000 time steps during which the end-effector had to stay in place (here, the control torques need to compensate for the swinging dumb-bell). In both tasks, a movement started and ended with zero velocity and acceleration. The velocity profile was sinusoidal with one peak.

For each task, three trajectories were pre-computed: eight-figures of three different sizes (Fig. 1 shows the big eight; small and medium eight are 0.9 and 0.95 of the big-eight's size, see Fig. 11) and three lines of different heights (Fig. 13). For training, data points were used from the two extremal trajectories, excluding the middle trajectory. For testing, all three trajectories were used.

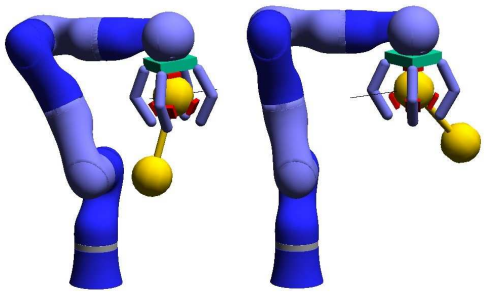


Figure 5: Robot swinging a dumb-bell. The black line shows the desired trajectory.

6.4 Control Architecture

We used an adaptive controller and separated training and test phases. To generate training patterns, a proportional-integral-derivative (PID) controller provided the joint torques. The proportional gain was chosen to be sufficiently high such that the end-effector was able to follow the desired trajectories. The integral component was initialized to a value that holds the arm against gravity (apart from that, its effect was negligible).

For testing, a composite controller provided the joint torques (Fig. 6). The trained feed-forward controller was put in parallel with a low-gain error feedback (its PD-gain was 1% of the gain used for training in the ball case and 4% in the dumb-bell case). For those feed-forward mappings that require an estimate of the object's mass, this estimate was computed using a Kalman filter as described above (the transition

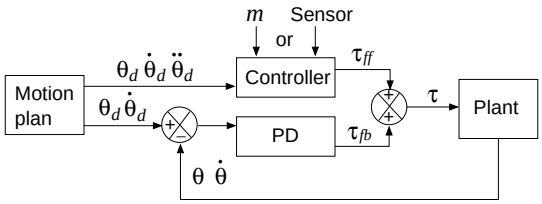


Figure 6: Composite control for the robot arm. A feed-forward controller is put in parallel with error feed-back (low PD gain).

noise Ω of m was set to 10^{-9} for both dynamics and sensor case).

6.5 Controller Learning

The feed-forward controller was trained on data collected during pure PID-control. For each mass context, 10 000 data points were collected in the ball case and 12 000 data points in the dumb-bell case. Half of these points (every second) were used for training and the other half for testing the regression performance. Three types of mappings were learned. The first maps the state and acceleration values $(\theta, \dot{\theta}, \ddot{\theta})$ onto joint torques. The second maps the same input onto the five sensory values, and the third maps the sensor augmented input $(\theta, \dot{\theta}, \ddot{\theta}, s)$ onto joint torques. The first two of these mappings were trained on two different labeled masses ($m_1 = 0.005$ and $m_2 = 0.03$ for ball or $m_2 = 0.06$ for dumb-bell). The last mapping used the data from 12 different mass contexts (m increased from 0.005 to 0.06 in steps of 0.005); here, the contexts were unlabeled.

Our learning task requires a non-linear regression technique that provides error boundaries for each output value (required for the Kalman-filtering step). Among the possible techniques, we chose locally-weighted projection regression (LWPR)[2] (Vijayakumar et al., 2005) because it is fast – LWPR is O(N), where N is the number of data points; in contrast, the popular Gaussian process regression is O(N^3) if making no approximation (Rasmussen and Williams, 2006).

7 RESULTS

The results of the robot-simulation experiments are separated into ball task – inference and control under varying mass – and dumb-bell task – inference and control under multiple varying context variables.

7.1 Inference and Control Under Varying Mass

For only one context variable, namely the mass of the manipulated object, both inference strategies (Sections 3 and 4) allowed to infer the unknown mass accurately (Figs. 7 to 10).

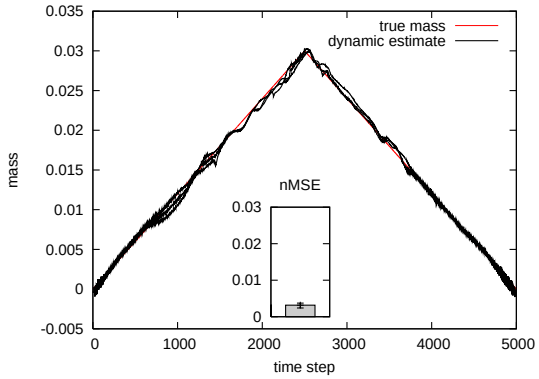


Figure 7: Inferring mass purely from dynamics. The inference results are shown for all three trajectories. The inset shows the normalized mean square error (nMSE) of the mass estimate. The error bars on the nMSE are min and max values averaged over an entire trajectory.

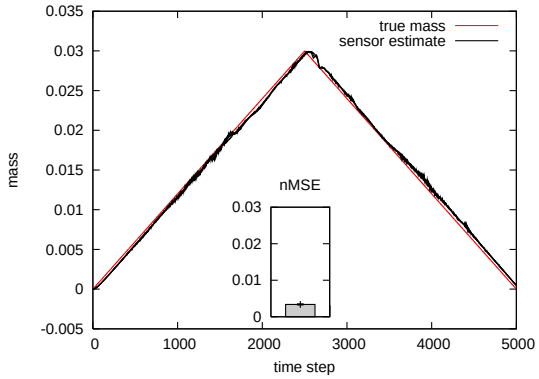


Figure 8: Inferring mass using tactile sensors. For details see Fig. 7.

Both types of mappings from state and acceleration either onto torques or onto sensors could be learned with low regression errors, which were of the same order (torques with m = 0.005: normalized mean square error (nMSE) = $2.9 * 10^{-4}$, m = 0.03: nMSE = $2.7 * 10^{-4}$; sensors with m = 0.005: nMSE = $1.3 * 10^{-4}$, m = 0.03: nMSE = $2.2 * 10^{-4}$). The error of the inferred mass was about the same for dynamics and sensor pathway. However, the variation between trials was lower in the sensor case.

Given the inferred mass via the torque and sensory pathways (Sections 3 and 4), our controller could provide accurate torques (Fig. 11). The results from both pathways overlap with the desired trajectory. As illustrated in the figure, a PID controller with a PD-gain as low as the gain of the error feed-back for the com-

[2] LWPR uses locally-weighted linear-regression. Data points are weighted according to Gaussian receptive fields. Our setup of the LWPR algorithm was as follows. For each output dimension, a separate LWPR regressor was computed. The widths of the Gaussian receptive fields were hand-tuned for each input dimension, and these widths were kept constant during the function approximation. Other learning parameters were kept at default values.

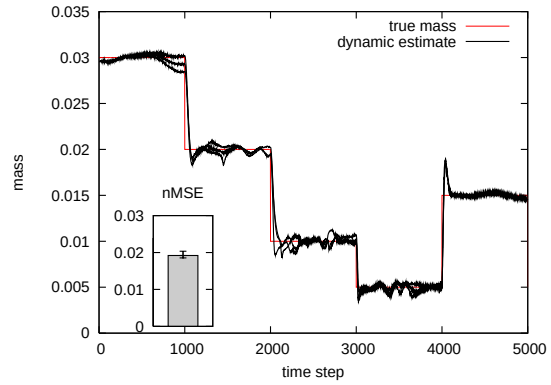


Figure 9: Inferring mass purely from dynamics. For details see Fig. 7.

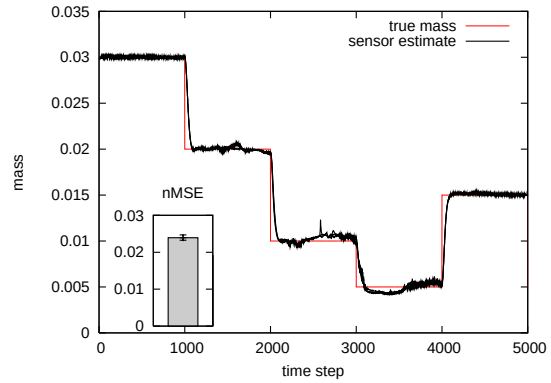


Figure 10: Inferring mass using tactile sensors. For details see Fig. 7.

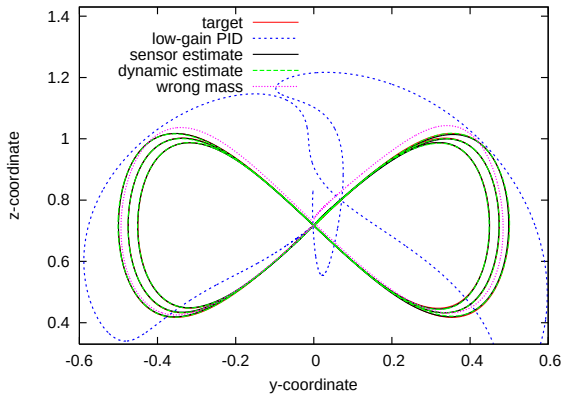


Figure 11: Following-the-eight task. The figure compares low-gain PID control (on large eight only) with the composite-controller that uses either the predicted mass estimate (from dynamics or sensors) or a wrong estimate ($m = 0.03$, on large eight only). The true mass decreased continuously from 0.03 to 0. The results for composite control based on the predicted mass overlap with the target for all three test curves.

posite controller was insufficient for keeping the ball on the eight figure. The figure furthermore illustrates that without a correct mass estimate, tracking was impaired. Thus, correctly estimating the mass matters for this task.

7.2 Inference and Control Under Multiple Varying Context

The inference of mass based on a single hidden variable failed if more variables varied (Fig. 12). We demonstrate this failure with the swinging dumb-bell, whose mass increased continuously during the trial. In the last part of the trial, when the dumb-bell was heavy and swung, the mass inference was worst. The wrong mass estimate further impaired the control of the robot arm (see Fig. 13).

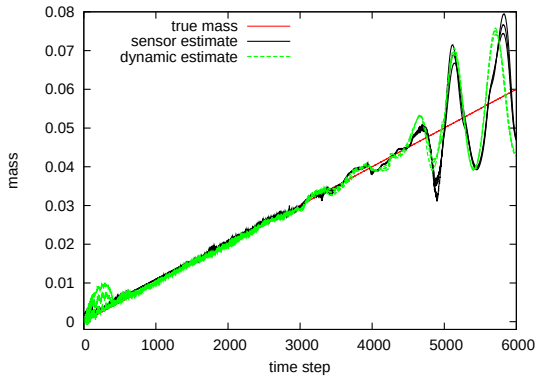


Figure 12: Inference of mass in the dumb-bell task.

During the last part of the movement, tracking was better with the direct mapping from $(\theta, \dot{\theta}, \ddot{\theta}, s)$ onto torques. For this mapping, the results still show some deviation from the target. However, we expect this error to reduce with more training data.

8 CONCLUSIONS

We presented three strategies for adaptive motor control under continuously varying context. First, hidden inertia parameters of a manipulated object can be inferred from the dynamics and in turn used to predict control torques. Second, the hidden mass of an object can be inferred from the tactile forces exerted by the object. Third, correct control torques can be predicted by augmenting the input $(\theta, \dot{\theta}, \ddot{\theta})$ with tactile forces and by directly mapping this input onto the torques.

We demonstrated the first two strategies with object mass as the varying context. For more context

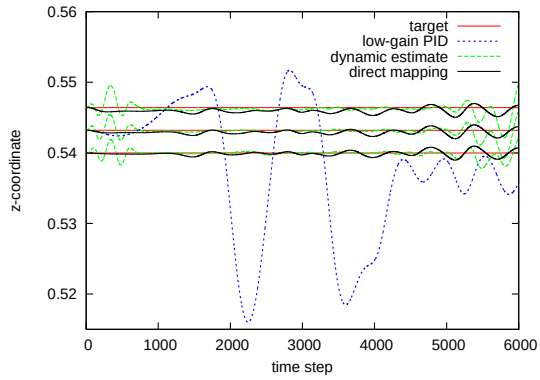


Figure 13: Swinging-the-dumb-bell task. The figure compares low-gain PID control (shown only on the middle trajectory) with the composite-controller that uses either the predicted mass estimate from dynamics or a direct mapping from $(\theta, \dot{\theta}, \ddot{\theta}, s)$ onto torques. For the estimate from sensors, the results were similar to the dynamics case and are omitted in the graph to improve legibility. The true mass increased continuously from 0 to 0.06.

variables, inferring the mass failed, and thus, the control torques were inaccurate. In principle, all inertia parameters could be inferred from the dynamics, but this inference requires modeling a 10-dimensional latent-variable space, which is unfeasible without extensive training data.

On the other hand, the direct mapping onto torques with sensors as additional input could predict accurate control torques under two varying context variables and in principle could cope with arbitrary changes of the manipulated object (including external forces). Further advantages of this strategy are its simplicity (it only requires function approximation), and for training, no labeled contexts are required.

In future work, we try to replicate these findings on a real robot arm with real tactile sensors. Real sensors might be more noisy compared to our simulated sensors; particularly, the interface between sensor and object is less well controlled.

ACKNOWLEDGEMENTS

The authors are grateful to the German Aerospace Center (DLR) for providing the data of the Light-Weight Robot III and to Marc Toussaint and Djordje Mitrovic for their contribution to the robot-arm simulator. This work was funded under the SENSOPAC project. SENSOPAC is supported by the European Commission through the Sixth Framework Program for Research and Development up to 6 500 000 EUR (out of a total budget of 8 195 953.50 EUR); the SENSOPAC project addresses the “Information Society Technologies” thematic priority. G. P. was funded by the Greek State Scholarships Foundation.

REFERENCES

Haruno, M., Wolpert, D. M., and Kawato, M. (2001). Mosaic model for sensorimotor learning and control. *Neural Computation*, 13:2201–2220.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Transactions of the ASME - Journal of Basic Engineering*, 82:35–45.

Narendra, K. S. and Balakrishnan, J. (1997). Adaptive control using multiple models. *IEEE Transactions on Automatic Control*, 42(2):171–187.

Narendra, K. S. and Xiang, C. (2000). Adaptive control of discrete-time systems using multiple models. *IEEE Transactions on Automatic Control*, 45(9):1669–1686.

Petkos, G., Toussaint, M., and Vijayakumar, S. (2006). Learning multiple models of non-linear dynamics for control under varying contexts. In *Proceedings of the International Conference on Artificial Neural Networks*. Springer.

Petkos, G. and Vijayakumar, S. (2007). Context estimation and learning control through latent variable extraction: From discrete to continuous contexts. In *Proceedings of the International Conference on Robotics and Automation*. IEEE.

Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*. MIT Press.

Sciavicco, L. and Siciliano, B. (2000). *Modelling and Control of Robot Manipulators*. Springer.

Vijayakumar, S., D’Souza, A., and Schaal, S. (2005). Incremental online learning in high dimensions. *Neural Computation*, 17:2602–2634.

Wolpert, D. M. and Kawato, M. (1998). Multiple paired forward and inverse models for motor control. *Neural Networks*, 11:1317–1329.

DEFECT-RELATED KNOWLEDGE ACQUISITION FOR DECISION SUPPORT SYSTEMS IN ELECTRONICS ASSEMBLY

Sébastien Gebus and Kauko Leiviskä
Control Engineering Laboratory, Department of Process and Environmental Engineering
University of Oulu, P.O. Box 4300, FIN-90014 Oulu, Finland
Firstname.surname@oulu.fi

Keywords: Decision support system, knowledge acquisition, quality, optimization, traceability, feedback.

Abstract: Real-time process control and production optimization are extremely challenging areas. Traditional approaches often lack in robustness or reliability when dealing with incomplete, inaccurate, or simply irrelevant data. This is a major problem when building decision support systems especially in electronics manufacturing, where blind feature extraction and data mining methods on large databases are common. Performance of these methods can be drastically increased when combined with knowledge or expertise of the process. This paper describes how defect-related knowledge on an electronic assembly line can be integrated in the decision making process at an operational and organizational level. It focuses in particular on the acquisition of shallow knowledge concerning everyday human interventions on the production lines, as well as on the conceptualization and factory wide sharing of the resulting defect information. Software with dedicated interfaces has been developed for that purpose. Semi-automatic knowledge acquisition from the production floor and generation of comprehensive reports for the quality department resulted in an improvement of the usability, usage, and usefulness of the decision support system.

1 INTRODUCTION

A decision support system (DSS) can be defined as "an interactive, flexible, and adaptable computer-based information system, especially developed for supporting the solution of a non-structured management problem for improved decision making. It utilizes data, provides an easy-to-use interface, and allows for the decision maker's own insights" (Turban, 1995). Data, however, does not exist naturally in a factory; it has to be collected, stored, prepared, and eventually mined. Morover, it might be incomplete, inaccurate, or simply irrelevant to the problem that is being investigated thus leading to the inability of decision makers to efficiently diagnose many malfunctions, which arise at machine, cell, and entire system levels during manufacturing operations (Özbayrack & Bell, 2002). These difficulties might be overcome by taking into consideration knowledge about the environment, the task, and the user (Gebus, 2006).

Knowledge-based approach takes advantage of the fact that it is the people operating the process who are most likely to have the best ideas for its improvement. It is through the integration of these ideas into the problem solving approach that a solution for long term process improvement can be found (Seabra Lopes & Caraminha-matos, 1995). Additionally, as the use of knowledge and more generally qualitative information better explains the relationships between input process settings and output response, it is well indicated for improving the understanding and usability of DSS (Spanos & Chen, 1997). In this paper, we shall examine the possibility to integrate knowledge in general, and especially shallow knowledge, into the decision making process. Section 2 presents the problematic related to knowledge acquisition and knowledge-related improvements in man/machine interractions. Section 3 presents our contribution to that field through a case study that is followed by a discussion on the results and the conclusion.

2 KNOWLEDGE ACQUISITION

Unlike data, knowledge does exist naturally in the factory, but collecting and interpreting it constitutes a major issue when building knowledge-based DSS.

These tasks commonly carried out by a knowledge engineer are often referred to as the bottleneck in the expert system development (Feigenbaum & McCorduck, 1983).

First and main obstacle is the knowledge engineering paradox (Liebowitz, 1993). Knowledge and skills that constitute expertise in a particular domain is tacit. Furthermore, the more competent experts become, the less able they are to describe how they solve problems. Another contribution to the bottleneck is the lack of willingness to share knowledge. It is often said that knowledge is power and people can be reluctant to give up what makes them inexpendable (Verkasalo, 1995). Finally, knowledge availability constitutes another obstacle as experts are not always known and have little time to spare. Additionally, today's global working conditions make it hard reaching experts located at the other end of the world or across the street at the subcontractor's plant. Distributed decision making becomes therefore a major issue (Verkasalo, 1995).

Currently, face to face discussions are still the most widely used way of transferring knowledge as they have the ability to make tacit knowledge more explicit by allowing the expert to provide a context to his actions. But expert interviews and other manual techniques are not always possible and depend very much on the knowledge engineer's own understanding of the domain. The challenge in a global company is therefore to develop tools and methods that enable experts to be their own knowledge engineers. Three topics are commented upon here: knowledge representation, automatic knowledge extraction and the user interface.

2.1 Knowledge Representation

Experts reasoning is often incomplete and not suitable for machine processing. Creating the proper ontology is therefore an essential aspect of sharing and manipulating knowledge. Based on the notion that different problems can require similar tasks, a number of generic knowledge representations have been constructed, each having application across a number of domains (Holsapple et al., 1989). Common classes of knowledge representations are logic, semantic network, or production rules.

Computer programs can use forms of concept learning to extract from exemples structural descriptions that can support different kinds of reasoning (MacDonald & Witten, 1989). More generally, automatic elicitation of knowledge, if possible, offers great advantages in terms of knowledge database generation.

2.2 Automatic Knowledge Extraction

Automatic knowledge extraction methods make it possible to build a knowledge base with no need for a knowledge engineer and only very little need for an expert, for example by using case-based reasoning. This poses however a knowledge acquisition dilemma: If the system is ignorant, it cannot raise good questions; if it is knowledgeable enough, it does not have to raise questions. Scalable acquisition techniques such as interview metasystems (Kawaguchi et al., 1991) or interviewing techniques using graphical data entry (Gaines, 1993) can help overcoming this difficulty.

Because the domain knowledge is often very specific, knowledge acquisition is a labor-intensive task. For that reason, generic acquisition shells have been developed (Chien & Ho, 1992) and extended with methods for updating incomplete or partially incorrect knowledge bases (Tecuci, 1992) (Su et al., 2002). The work has also been facilitated by studies on the automatic acquisition of shallow knowledge, which is the experience acquired heuristically while solving problems (Okamura et al., 1991), or by compensating for the knowledge engineer's lack of domain knowledge, so that the resulting knowledge base is accurate and complete (Fujihara et al., 1997).

2.3 User Interface

In DSS users are often presented with an exhaustive amount of data upon which they have to make decision without necessarily having the proper understanding or knowledge to do so. The user interface (UI) is the dialogue component of a DSS that facilitates information exchange between the system and its users (Bálint, 1995).

The choice of an interface depends on many factor, but there are only few reasons for its inadequacy (Norcio & Stanley, 1989). Mainly, the UI is often seen as the incidental part of the system. Consequently it is not well suited to the system or to the user, and more often to neither. Usability can be seen as the degree to which the design of a particular UI takes into account the psychology and physiology of the users, and makes the process of using the system effective, efficient and satisfying.

For its response to be understandable, a DSS should be able to tailor its response to the needs of the individual. UI adaptability can be achieved by mapping user's actions to what they intend to do (Eberts, 1991) or need (Lind et al., 1994). This can undermine however the user's confidence in the information given to him. Adaptability can therefore

be applied to the content instead of the interface itself by increasing the ability of a DSS to explain itself for example by using graphical hierarchies instead of the equivalent flat interface to describe the structure of a rule base (Nakatsu & Benbasat, 2003).

Different planning and design methodologies have been developed to insure that user specifications are taken into consideration (Wills, 1994) (Balasubramian et al., 1998) (McGraw, 1994). The following steps, also used in the case study, aim to build a separate methodology to develop user interfaces for knowledge acquisition:

- Identify and characterize the real users;
- Define a work process model;
- Definition of a general fault model;
- Design of a prototype, and
- Test, debugging, and redesign.

3 CASE STUDY

The electronic subcontractor involved in this case study is lacking the resources for long-term process improvement. Process control is left to the operators who make adjustments only based on experience and personal knowledge of the production line. Thus tuning of the system and consequent quality of the product depend very much on human interpretation of machine problems. A DSS integrating this "know-how" could lower the variability inherent to human choices and greatly improve the efficiency of any response when a problem occurs.

The proposed DSS tries to provide understanding and formalization of the parameters influencing the quality of the products, which are needed to improve the operative quality. Traceability in terms of know-how from the production floor is achieved through an Expert Knowledge Acquisition System (EKCS) recording the main breakdown information. Cross-analysis of the subsequent enriched information with measurement data can then improve the ability to control the production as described in Figure 1.

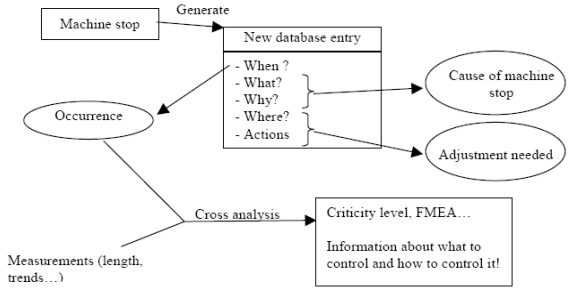


Figure 1: Flowchart representing the Expert Knowledge Acquisition System.

3.1 Prototype Version

The approach based on fast prototyping has already been described (Gebus, 2006). It is well suited for creating tools that are both user and context specific. This section serves as a reminder of the main points and conclusions concerning the prototype version.

Line operators are the main users and knowledge providers of the DSS, whereas quality engineers are interested in getting knowledge in a way, which is most suitable for a comparative analysis with measurement data. The process model describes the interactions between the process and the knowledge sources. The fault model analyzes the cause and effect relationships in the process model. Finally, a prototype interface is designed to guide the operator through the knowledge acquisition process using pictures of the product and check boxes representing the different cells of the production line. Fault information is entered manually.

Prototype versions are time consuming but have the advantage to allow the identification of problems that the knowledge engineer would otherwise be unaware of. All these problems can then be addressed when designing a more complete version.

3.2 Final Version

3.2.1 Axes of Improvement

Based on the feedback from the test period, three different user-oriented axes of improvement were identified, Usability, Usefulness, and Usage (3U).

Usability concerns mainly knowledge input that needs to be simpler, faster, and more intuitive. The input method also has to favor system portability as equipment is in constant evolution. In this context, checkboxes that affect the appearance of the UI are confusing and are replaced by clickable areas on digital photographs of the production line and cells. A defect is located by zooming in on an area, which is linked to a list of problems and corrective actions. This approach enables the needed flexibility and portability while keeping the environment and the interface very familiar, resembling a factory floor.

Usefulness probably is the most important target of any system. However, experience has shown that a system aimed at simplifying operators' work, but updated by design engineers, does not have any long-term continuity. Motivating every single user of the system into actually using is achieved by transforming the DSS from a simple fault collecting system to a factory wide information sharing system providing user-specific levels of added value.

Improved usage is consequent to improved usefulness. Defect information is sent back to the operators in forms that can be used during meetings to discuss encountered production problems. In the same way, quality engineers are more inclined to use a system that automates some of their tasks and provides information tailored to their needs it.

3.2.2 Overall Structure

In addition to specific interfaces for operators and quality engineers, a dedicated UI has been created for updating the system. As an information sharing system benefiting all the users, Figure 2 shows only closed loops of information flows. Operators provide defect information stored in a database. The quality department can access any relevant historical data to produce statistical information. After analysis, quality feedback is generated and sent back to the production line. An administrator uses defect information only casually for updating the system.

From the practical point of view, the three interfaces use a unique database allowing an automatic and immediate update of the system. This database is stored on a SQL server providing the needed flexibility that was missing in the prototype version. Such a structure enables storing not only information about date, time, defect and corrective actions, but also all the settings relative to a certain production line. This was added in order to make the administrator interface a fully integrated subsystem.

3.2.3 Some Features in Details

The administrator interface was designed so that new setups can be created easily. Updating is done by selecting digital photographs of the production lines and cells, and creating clickable areas that are then linked to defect types and possible solutions.

Concerning the operator interface, emphasis has been put on simplicity and intuitiveness. Setup options are limited to choosing the database location and selecting the current line. This is done only once when the DSS is implemented on a new production line or when the digital photographs are updated. In normal use, data input is done by choosing the product part from a list, and choosing the defect area by zooming in on the pictures as in Figure 3. The list of known causes and corrective actions is made available as well as a comment window in case of unreferenced problems. Selection of causes and actions generates automatically a decision tree and updates charts representing short-term quality and usage information.

The supervisor interface is more complex and offers three main options as shown in Figure 4. Similar reporting capability as in the operator interface is available, but with no limitations in time. Long-term trends can be visualized from historical data. Furthermore, advanced database exploration properties have been added. Customizable SQL requests can be created and run on the entire database before exporting the result to other software.

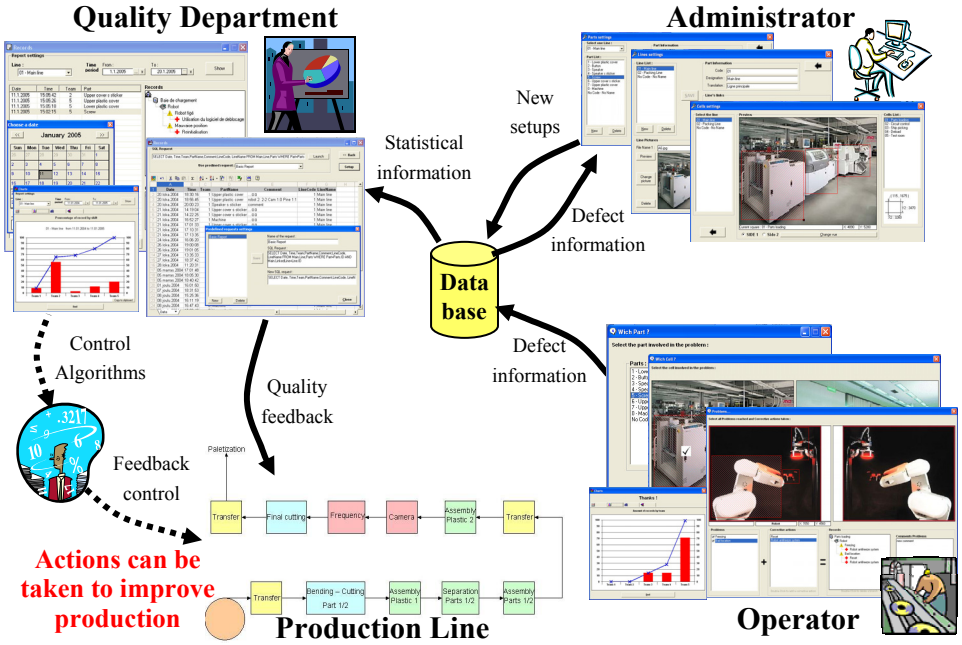


Figure 2: General structure of the DSS with information flows between the different entities.

4 RESULTS AND DISCUSSION

During the testing of the prototype version 183 machine stops occurred, out of which only 70 were commented on representing therefore a usage rate of 38%. After having made the modifications, the final version started to be used in a systematic way at different levels of the company, thus promoting employee involvement towards quality issues. Line operators, in particular, not only increased the usage rate to nearly 100%, but extend that use to their weekly quality meetings. For them the increased usage has been triggered by an improved usefulness.

The acceptation level for this new tool is based on fully graphical interactions and digital pictures. These enable quick updates of the system while keeping a familiar framework. It also shows that in order to satisfy the user needs, the real challenge for an adaptable interface is not to evolve with the problem, but rather to remain static while presenting an evolving situation.

The new systems enables the cross analysis of data and expert information, which is a prerequisite for developing feedback control policies that will lead to a more efficient factory-wide knowledge and defect management. The supervisor interface in particular can be the backbone for implementing monitoring and control algorithms. It has shown that any kind of previously stored data can not only be easily accessible, but also be processed by any chosen algorithm and the results can be sent back in various forms (charts, decision trees etc.).

One can also imagine replacing manual feedback with control algorithms generating automatic feedback control. This is not possible in the current state of the system as proper actuators necessary to transform information into action on the production line are missing. Even if they did exist, automatic feedback control would still be highly dependent on line technology and therefore not portable.

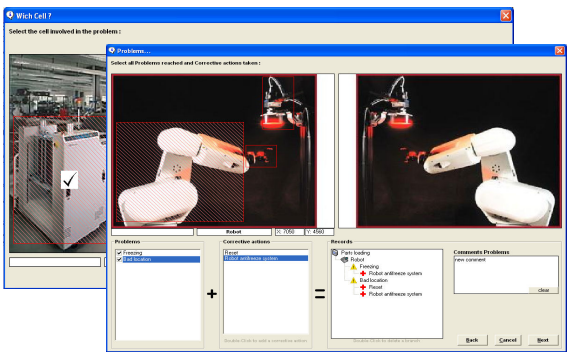


Figure 3: Interface for fault selection.

5 CONCLUSION

When building a knowledge-based system, the approach that is used has to be very human oriented. Defining the right interfaces for real-time knowledge acquisition can be a major problem. They have to be adapted to users with various degrees of knowledge. In addition to this, the complexity of any interface must be sufficient enough to catch the full scope of information, but simultaneously keep the data extraction process as simple as possible.

The general process for designing a knowledge acquisition interface applied to this case study presents the different tasks that have been undertaken and the problems encountered. Unlike traditional design techniques that emphasize doing it right the first time, the 3U approach proposed in this section leads to a better match with user concerns. Knowledge acquisition software has been implemented on the production floor in a factory producing components for the electronics industry. Based on a test period, the knowledge gained from the use of this tool enabled defect classification and standardization. This is the first step towards cross analysis with monitored parameters from the production floor, leading eventually to on-line fault diagnosis.

REFERENCES

Balasubramanian, V., Turoff M., Ullman, D., 1998, A Systematic Approach to Support the Idea Generation Phase of the User Interface Design Process. *31st Hawaii Int. Conf. on System Sciences*. vol. 6, pp. 425 – 434.

Bálint, L., 1995, Adaptive Interfaces for Human-Computer Interaction: A Colorful Spectrum of Present and Future Options. *IEEE Int. Conf. on Systems, Man and Cybernetics*. vol. 1, pp. 292 – 297.

Chien, C.C., Ho, C.S., 1992, Design of a Generic Knowledge Acquisition Shell. *IEEE Int. Conf. on Tools with AI*. pp. 346 – 349.

Eberts, R., 1991, Knowledge Acquisition Using Neural Networks for Intelligent Interface Design. *IEEE Int. Conf. on Systems, Man, and Cybernetics*. vol.2, pp. 1331 – 1335.

Feigenbaum, E.A., McCorduck, P., 1983, *The fifth generation: artificial intelligence and Japan's computer challenge to the world.* Addison-Wesley Longman Publishing Co.

Fujihara, H., Simmons, D.B., Ellis, N.C., Shannon, R.E., 1997, Knowledge Conceptualization Tool. *IEEE Transactions on Knowledge and Data Engineering* 9(2): 209 – 220.

Gaines, B.R., 1993, Eliciting Knowledge and Transferring it Effectively to a Knowledge-Based System. *IEEE Transactions on Knowledge and Data Engineering* 5(1): 4 – 14.

Gebus, S., 2006, *Knowledge-Based Decision Support Systems for Production Optimization and Quality Improvement in the Electronics Industry*. Oulu university press.

Holsapple, C.W., Tam, K.Y., Whinston, A.B., 1989, Building Knowledge Acquisition Systems – A Conceptual Framework. *22nd Hawaii Int. Conf. on System Sciences.* vol. 3, pp. 200 – 210.

Kawaguchi, A., Motoda, H., Mizoguchi, H.R., 1991, Interview-Based Knowledge Acquisition Using Dynamic Analysis. *IEEE Expert*: 47 – 60.

Liebowitz, J., 1993, Educating Knowledge Engineers on Knowledge Acquisition. *IEEE Int. Conf. on Developing and Managing Intelligent System Projects.* pp. 110 – 117.

Lind, S., Marshak, W., 1994, Cognitive Engineering Computer Interfaces: Part 1 – Knowledge Acquisition in the Design Process. *NAECon 1994 Conf.* vol.2, pp. 753 – 755.

MacDonald, B.A., Witten, I.H., 1989, A Framework for Knowledge Acquisition through Techniques of Concept Learning. *IEEE Transactions on Systems, Man, and Cybernetics* 19(3): 499 – 512.

McGraw, K.L., 1994, Knowledge Acquisition and Interface Design. Software, *IEEE* 11(6): 90 – 92.

Nakatsu, R.T., Benbasat, I., 2003, Improving the Explanatory Power of Knowledge-Based Systems: An Investigation of Content and Interface-Based Enhancements. *IEEE Transactions on Systems, Man, and Cybernetics* 33(3): 344 – 357.

Norcio, A.F., Stanley, J., 1989, Adaptive Human-Computer Interfaces: A Literature Survey and Perspective. *IEEE Transactions on Systems, Man, and cybernetics* 19(2): 399 – 408.

Okamura, I., Baba, K., Takahashi, T., Shiomi, H., 1991, Development of Automatic Knowledge-Acquisition Expert System. *Int. Conf. on Industrial Electronics, Control and Instrumentation.* vol.1, pp. 37 – 41.

Seabra Lopes, L., Camarinha-Matos, L.M., 1995, A Machine Learning Approach to Error Detection and Recovery in Assembly. *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems.* vol. 3, pp. 197 – 203.

Spanos, C.J., Chen, R.L., 1997, Using Qualitative Observations for Process Tuning and Control. *IEEE Transactions on Semiconductor Manufacturing* 10(2): 307 – 316.

Su, L.M., Zhang, H., Hou, C.Z., Pan, X.Q., 2002, Research on an Improved Genetic Algorithm Based Knowledge Acquisition. *1st Int. Conf. on Machine Learning and Cybernetics.* pp. 455 – 458.

Tecuci, G.D., 1992, Automating Knowledge Acquisition as Extending, Updating, and Improving a Knowledge Base. *IEEE Transactions on Systems, Man, and cybernetics* 22(6): 1444 – 1460.

Turban, E., 1995, *Decision support and expert systems: management support systems.* Englewood Cliffs, N.J., Prentice Hall.

Verkasalo, M., 1997, *On the efficient distribution of expert knowledge in a business environment.* Oulu university press.

Wills, C.E., 1994, User Interface Design for the Engineer. *Electro/94 International Conference.*

Özbayrak, M., Bell, R., 2003, A Knowledge-based decision support system for the management of parts and tools in FMS. *Decision Support Systems* 35(4): 487 – 515.

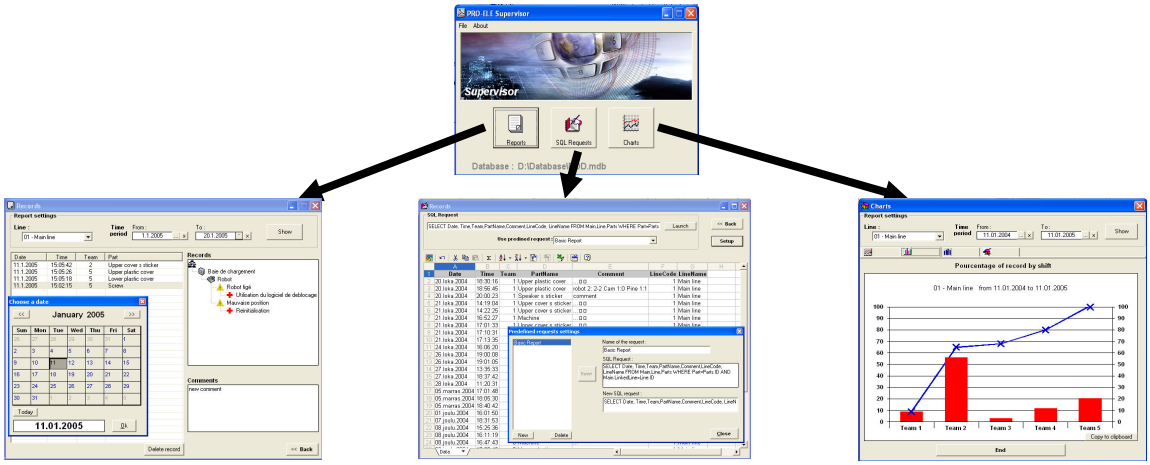


Figure 4: Structure of the supervisor interface.

A NEW LOAD ADJUSTMENT APPROACH FOR JOB-SHOPS

Z. Bahroun
LIP2, Faculté des Sciences de Tunis, Dép. des Sciences de l'Informatique, 2092 Manar II, Tunis, Tunisie
zied.bahroun@ fst.rnu.tn

J.-P. Campagne
Laboratoire LIESP, INSA-LYON, F-69621 Villeurbanne Bat B. Pascal, 7 av J. Capelle, France
jean-pierre.campagne@insa-lyon.fr

M. Moalla
LIP2, Faculté des Sciences de Tunis, Dép. des Sciences de l'Informatique, 2092 Manar II, Tunis, Tunisie
mohamed.moalla@fst.rnu.tn

Keywords: Load adjustment, Finite capacity, overlapping production planning, margins management, job-shop.

Abstract: This paper presents a new load adjustment approach by overlapping for a set of jobs in a job-shop context, guaranteeing the existence of a limited capacity schedule without scheduling under the assumption of pre-emptive tasks. This approach is based on the exploitation of the tasks scheduling time segments overlapping and on the distribution of the job's margins between tasks in a just in time context. First, we present a literature review concerning load adjustment approaches. Second, we introduce the overlapping load adjustment approach. Third, we present an original heuristic to use this approach in the case of job-shops organized firms. After that, we present the scheduling approach. Finally, we will discuss a more general use of this approach and the possible extensions.

1 INTRODUCTION

Generally, the production planning is made in a hierarchical way in two planning and scheduling decision levels. In the first step, we decide which products to supply, in which quantities and delays, in the second step, we adjust load to the capacity and schedule the tasks on the machines.

There are three main and classical load adjustment approaches. First of all, we have the placement. This approach consists in calculating a detailed tasks schedule. A new task is integrated in the planning if we find a gap in the planning which is bigger than the duration of this task. This approach estimates only one schedule which can be destructed by any disturbance. The second approach is the periodic and cumulative approach. It consists in calculating the cumulative load and capacity for a latest loading and for each period. This approach does not guarantee the existence of a scheduling solution because it does not take into consideration the ready dates constraints. The third approach is the periodic and non cumulative approach. It consists in assigning tasks to periods and comparing period by period the available and the required capacities. This method estimates only the solutions in which the tasks are fixed in a specific period.

Some researchers studied the problem of sequencing decisions in production planning and scheduling. Dauzere-Peres and Laserre (1999) think that it is better in some cases to integrate the scheduling decision level in the lot sizing decision level and propose an iterative approach for planning and scheduling production. Some researchers integrated the scheduling and capacity constraints in their lot sizing model (see for example, Fleishmann and Meyr, 1997). We can also find a survey on lot sizing and scheduling in Drexl and Kimms (1997). However, most of these approaches consider generally a single machine and are difficultly applicable for real and industrial context.

Many researchers studied also the problem of finite capacity planning. We can state very briefly H. Hillion and Proth (1994) who studied the problem of

a finite capacity flow control in a multi-stage/multi-product environment or Gunther (1987) who also developed two heuristics for the lot sizing in the context of finite capacity. Néron et al (2001) developed an approach for solving hybrid flow shop problem using energetic reasoning. This approach has some similarities in the concept with our approach. Indeed, the energetic reasoning was developed to solve "cumulative scheduling problems". The approach aims to develop satisfiability tests and time-bound adjustments to ensure that a given schedule is not feasible or derives some necessary conditions that any feasible schedule must satisfy.

In comparison with all these approaches, the overlapping load adjustment approach allows to distinguish two main phases. The first phase consists on establishing a long or mid term production planning where the feasibility is ensured without scheduling and tasks placement, which allows us to characterize a set of feasible scheduling solutions. The scheduling will be done only in the second phase.

2 THE OVERLAPPING LOAD ADJUSTMENT APPROACH

The time scale is divided into time periods. Each task of a job has got a processing time, requires one or more resources and has to be realized during a scheduling time segment associated with one or more consecutive periods. The scheduling time segments of consecutive tasks of the same job cannot overlap From now on and throughout this paper a lapse of time called here lapse, designates a succession of a number of n consecutive periods. Let (a,b) be a lapse composed of a succession of periods which are limited by the periods a and b including them. The shortest lapses are composed of only one period, for instance (a,a). Such a lapse (a,a) is called a basic lapse. The longest lapse is noted (1,H) in which number 1 is associated with the first period of the planning time frame and the letter H the last one. From such a planning time frame, the total number of different lapses is equal to H*(H+1)/2.This number is of course to be multiplied by the number of existing processors. The sub-lapse of a lapse is a subset of one or more consecutive periods of this lapse. For instance, the sub-lapses of [1,3] are [1,1], [2,2], [3,3], [1,2] and [2,3]. Every lapse containing a lapse [a,b] is called the over-lapse of [a,b]. For instance, [1,3] is an over-lapse of [1,1]. Each lapse is characterized by:

- An accumulated capacity: sum of the capacities of each period included in this lapse.
- A direct capacity requirement: sum of the capacities required by the tasks whose scheduling time segment is exactly equal to this lapse.
- An accumulated capacity requirement: sum of the capacities required by the tasks whose scheduling time segment is fully included in this lapse. It is the sum of the direct capacity requirements of this lapse and its sub-lapses.

Dillenseger (1993) sets the following proposition out: for any lapse, its accumulated capacity requirement must be equal or smaller than its accumulated capacity. He proves that it is a necessary and sufficient condition for the existence of a loading solution of the set of tasks (within the limits of their scheduling time segments and considering the capacity levels), according to pre-emptive possibility.

Let's consider the following example of a production plan composed of 7 jobs which will be treated on a single processor with ready and due dates as shown in Table 1 below. The considered period for this example is the week composed of five days (the day is the unit time):

Table 1: Example 1.

Job	processing time (days)	Ready date (beginning of the week)	Due date (ending of the week)
A	2	3	4
B	4	2	4
C	4	4	5
D	1	3	3
E	2	3	3
F	2	2	3
G	4	4	5

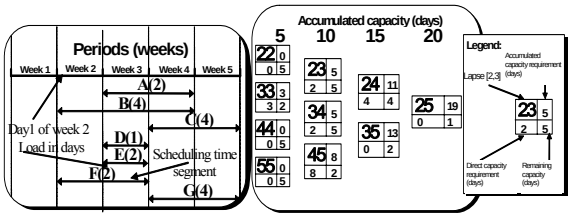


Figure 1: Capacity requirements planning (CRP) and Planning feasibility control graph (PFCG).

Figure 1 represents a capacity requirement planning (CRP) corresponding to the example of Table 1. For instance, task G needs a load of 4 days

(time units) and have a time scheduling segment composed of weeks 4 and 5 (it means that this task should be scheduled and produced in any time inside the weeks 4 and 5). The capacity of a period is 5 days. The margin of task G is so equal to 6 days (10 days of weeks 4 and 5 minus its load of 4 days). The associated planning feasibility control graph (PFCG) is shown in Figure 1. For each lapse of weeks we calculate the direct capacity requirement, the accumulated capacity requirement and the remaining capacity. For instance, the lapse [2, 3] composed of weeks 2 and 3 has a direct capacity requirement equal to 2 days (task F), an accumulated capacity requirement equal to 5 days (tasks with a scheduling time segment included in the lapse: D, E and F) and a remaining capacity equal to (10-5) days. The planning feasibility control graph proves the feasibility of this set of jobs (all the remaining capacities are positive).

Firstly, this load adjustment approach was applied to plan the activities of a make-to-order company in a mono-level context (Dillenseger, 1993). This approach was applied then to a flow-shop composed of m processors (Bahroun, 2000a), to a generalised flow-shop (Bahroun, 2000b) and for the cyclic production context (Bahroun, 1999).

3 APPLICATION FOR JOB-SHOP ORGANIZED FIRMS

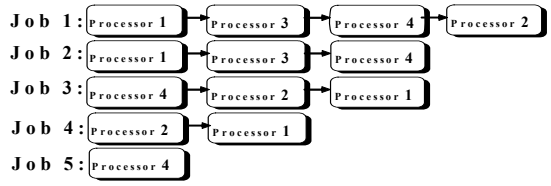


Figure 2: Example of a job shop.

Let us consider N jobs with their due and ready dates. Each job is composed of one to m tasks realized on one to m processors with a certain order which is not necessarily the same for all the jobs (Figure 2).We suppose that these jobs are the results of a products supply calculation in an M.R.P. based system for instance. **We aim at adjusting the load resulted** by these jobs in a **finite capacity way** by adapting the overlapping load adjustment approach to the job shop case.

We will first calculate the scheduling time segment of each task, considering the due and the ready dates of their job, their precedence constraint and the capacity constraints. After that, we will try to exploit the existing margins. They will be distributed on the different job's tasks and will be assigned with priority to the tasks corresponding to overloaded processors. We have developed a heuristic which tries to share out judiciously the job's margins on their tasks. For this, we calculate a latest loading on all the processors without margins (we only assign the margins of jobs with a unique task). We classify in the load decreasing order the processors. We then assign all the margins to the most loaded processor and after that we keep only the necessary margins to validate the processor loading (for any lapse, its accumulated capacity requirement must be equal or smaller than its accumulated capacity) and transfer the unused margins to the next processor accordingly to the load classification. Then, we reiterate the same treatment to the next processor until reaching the last processor.

We define the following parameters:

- N = number of jobs
- m = number of processors
- Ma_i = the global margin of job i
- p_{ij} = processing time of the task corresponding to job i on processor j.
- p_{ij} = 0 if there is not a task of job i on processor j.
- r_i = release or ready date of job i
- d_i = due date of job i
- b_{ij} = beginning of the scheduling time segment of the task corresponding to job i on processor j
- e_{ij} = ending of the scheduling time segment of the task corresponding to job i on processor j
- $ACCP[a,b]_j$ = accumulated capacity of the lapse [a,b] for processor j
- $ACCPR[a,b]_j$ = accumulated capacity requirement of the lapse [a,b] for processor j
- $RC[a,b]_j$ = remaining capacity of the lapse [a,b] on processor j
- d_p = duration of an elementary period.

We note $\lceil x \rceil$ the smallest integer which is greater than or equal to x and $\lfloor x \rfloor$ the biggest integer which is smaller than or equal to x.

Our approach is based on four main steps (we will illustrate our approach with the example of Table 2):

1st step :

We calculate the global job's margins:

$$Ma_i = d_i - r_i - \left(\sum_{j=1}^{m} \lceil (p_{ij} / d_p) \rceil\right) + 1 \qquad (1)$$

Table 2: Exemple 2.

Job	Processing order	Process. time on proc. 1 (days)	Process. time on proc.2 (days)	Process. Time on proc. 3 (days)	ready date (beginning of the period)	due date (ending of the period)	Global margins in periods (weeks)
A	P1→P2→P3	**3**	**2**	**2**	1	6	**3**
B	P1→P2→P3	**2**	**3**	**1**	1	5	**2**
C	P1	**3**	0	0	3	5	**2**
D	P2	0	**2**	0	2	4	**2**
E	P3	0	0	**2**	4	6	**2**
F	P3→P2→P1	**3**	**3**	**2**	2	5	**1**
G	P3→P2→P1	**1**	**1**	**1**	2	6	**2**
H	P3→P1→P2	**2**	**2**	**2**	3	5	**0**
I	P3→P1→P2	**1**	**2**	**2**	3	5	**0**
J	P2→P1	**3**	**3**	0	2	5	**2**
K	P2→P3	0	**3**	**4**	3	6	**2**
L	P3→P1	**2**	0	**3**	4	6	**1**

For our example, we consider a production system composed of three processors (P1, P2 and P3) and a set of jobs (Table 1, d_p = 5 days). We calculate the global margins using the last formula and we obtain the results reported in the last column of Table 2. After that, we determine the scheduling time segment of each task according to a latest loading without margins (we assign only the margins of jobs with a unique task like jobs C, D and E, in fact, these margins will be used only on a unique processor and will not be distributed on several processors). For instance, for job A, the last task on processor P3 will have a scheduling segment that ends at the end of week 6 and will begin so, at the beginning of the same week (because the processing time of this task is inferior to a week), the task number 2 for the same job A on processor P2 will have a scheduling segment that begins and ends at week 5. The first task of Job A on processor P1 will have a scheduling segment that ends and begins at week 4. Job C is composed of only one task, so we'll assign its margin and the time scheduling segment will begin at week 3 and ends at week 5. We calculate the scheduling segments of the other jobs in the same way and we obtain the capacity requirement planning of P1, P2 and P3 as shown in Figures 3 and 4.

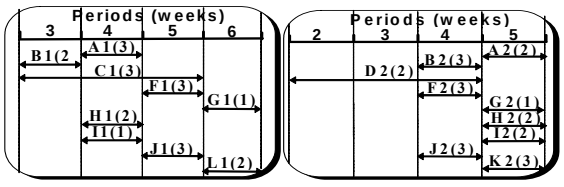


Figure 3: Capacity requirements planning (CRP) of P1 and P2 before treating.

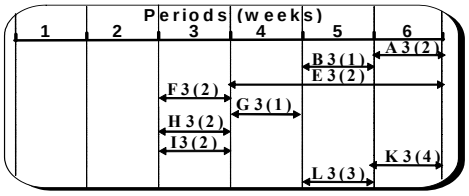


Figure 4: Capacity requirements planning (CRP) of P3 before treating.

After that, we calculate the total load and the load by period for each processor as follow:

Table 3: Calculation of the processor load.

Proces.	Total load in days	Number of concerned Periods(weeks)	Load/period
P1	20	4	5
P2	21	4	5,25
P3	19	4	4,75

We classify and treat the processor in the decreasing order of the load/period: P2, P1, P3.

2nd step:

We assign all the global margins to the processor P2. Then, we calculate for each task the beginning and the ending periods of the scheduling time segment of this task:

∀ i and for a processor j:

e_{ij} remains the same

$$b_{ij} = e_{ji} - Ma_i - \lceil (p_{ij} / d_p) \rceil + 1 \qquad (2)$$

If we calculate the beginning time of the second processor of our example, we can generate the

corresponding capacity requirement planning (CRP) and the planning feasibility control graph (PFGC):

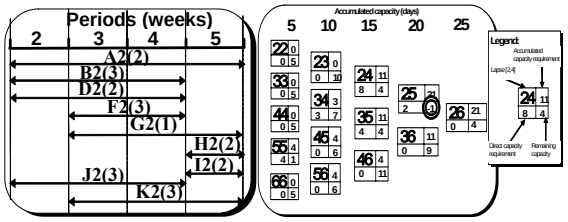


Figure 5: CRP and PFCG of the processor P2 after assigning all the margins.

3rd Step:

We then check the feasibility condition. If we find problems in certain lapses (even with all available margins), we should lengthen the scheduling time segment of some tasks. If for a lapse [a,b] in the processor j, the validation condition is not verified:

- We consider in the duration increasing order all the tasks that $b_{ij} \geq a$ et $e_{ij} = b$.
- In this list, we begin to treat the first task in the k^{th} position in the list with a load equal or superior to the overloading in the lapse which allows us to delay the minimum number of tasks.
- We proceed progressively to the lengthening of the scheduling time segment of these tasks, period by period until the verification of the feasibility condition or arrival to the end of the list.
- If we arrive to the end of the list we try with tasks positioned in the $(k-1)^{th}$, $(k-2)^{th}$ … position until the verification of the feasibility condition.

In our example (Figure 5), we remark that we have an overloading in the lapse [2,5] which obliges us to delay of one period, one of the tasks included in this lapse and which finishes in period 5 (A2, G2, H2, I2 or K2 as we can see in CRP of Figure 5). We choose in this example to lengthen the scheduling time segment of task G2 from the lapse [3,5] to the lapse [3,6].

Remark: If we do not accept to delay jobs, the lapses with negative remaining capacity indicate where we must increase the capacity by using for example overtime or interims. We can also introduce the notion of jobs priority for choosing which tasks must be delayed.

4th Step:

Now, we will try to regain margins. We begin with the tasks corresponding to jobs with weak global margins. Tasks of jobs without margins are assigned to the elementary lapses ([1,1], [2,2] etc.), those corresponding to jobs with one period margin are assigned to the lapses of the second column of the feasibility control graph, those corresponding to jobs with k periods margins are assigned to the column number k etc. Our treatment begins with the lapses of the second column because the corresponding jobs have only one period global margin and we must preserve these precious margins to validate the other processors and use the margins from jobs that have important global margins.

A transfer of a task from the lapse [a,b] to the lapse [a+w,b], adds load to all the over-lapses of [a+w,b] which are not initially over-lapses of [a,b]. The transferred load must be equal or smaller than the remaining capacity on these lapses for maintaining the validation condition.

The proposed approach for this transfer tries to transfer the maximum number of tasks and tries to match in the best way the transferred load in regard to the remaining capacity. Consequently, we construct the set of tasks which can be transferred, and we classify this set in the increasing order of their load. We transfer the tasks one by one in this order while the sum of their load is smaller than the remaining capacity. Then, we take the last task transferred and we try to change it by another task from the remaining tasks of the set and which matches better the remaining capacity. If two tasks have the same load we can choose for example the task corresponding to a product with a greater carrying cost. If we take the example of PFCG in Figure 5, we begin with the lapses of the second column. If we try, for instance, to regain margins from the tasks corresponding to the lapse [3,4], we should transfer the maximum number of tasks to the lapse [4,4]. We can transfer task F2 because the minimum of the remaining capacity of the over-lapses of [4,4] which are not over-lapses of [3,4] (the lapses [4,4], [4,5] and [4,6]) is 5 and it is greater than the load of the task F2. Then, we pass to lapses [4,5], [5,6], and next to the lapses of the third column (for the task G2, we succeed to regain 2 periods, the scheduling time is shortened from the lapse [3,6] to lapse [5,6]). We reiterate this treatment until arriving to the last column. We obtain after treatment of the processor P2 the capacity requirement planning of Figure 6.

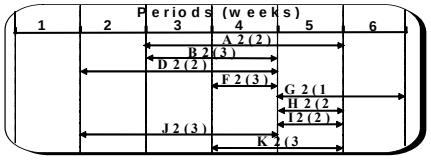


Figure 6: C.R.P. of processor P2 after treatment.

We can generalize this step for a processor j:

$\forall$ the lapse [x,y] / x= a and y= a+w with $1 \le w \le H$ and $1 \le a \le H-w$:

- Construct the set A = {set of tasks / b_{ij} = a and e_{ij} = a+ w} (The set of tasks that their scheduling time segment can be shortened, this set will be ordered in the increasing order of their load).
- Iterate then for f = w -1 $\rightarrow$ 0
  - Cap = Min (RC_j [a+1+z, a+w+n]) For n = 0 $\rightarrow$ H - a - w, z = f $\rightarrow$ 0 (we calculate the maximum load that can be transferred).
  - Transfer the maximum number of tasks from A (using the approach described in the precedent page) that the sum of their processing times is inferior or equal to Cap. Let C be the set of these tasks transferred and Q the sum of their processing time:

    $\forall$ the task ij $\in$ C, we put b_{ij} = a+1 + f

    For n = 0 $\rightarrow$ H - a - w and for z = f $\rightarrow$ 0, we do:

    RC_j [a+1+z, a+w+n] = RC_j [a+1+z, a+w +n] – Q (We update the new time scheduling segment of the tasks and the remaining capacity of the concerned lapses).
- We update the beginning and the ending of the scheduling time segments of the other tasks on the other processors as follow:
  - If a task u precedes task j of the same job i on the processor j, we move its scheduling time segment in a manner that the ending time becomes equal to the beginning time of the task j. We effectuate the same treatment until arriving to the first task.
  - If a task u follows task j of the same job i on the processor j, we update in a symmetrically manner the scheduling time segment of this task and all the other tasks up to the last one.

5th step:

We assign all the unused margins to the next processor (in this case the processor P1). We calculate the scheduling time segments of the tasks corresponding to this processor using the following formulae:

$\forall$ i , for a processor j :

$$b_{ji} = e_{ji} - M'a_i - \lceil (p_{ij} / d_p) \rceil + 1 \qquad (3)$$

Where $M'a_i$ is the remaining margin.

We can assign margins for a task i on a processor j only if the precedent tasks of the same job are not already treated. We obtain the C.R.P. and the control feasibility graph of the processor P1 as follow:

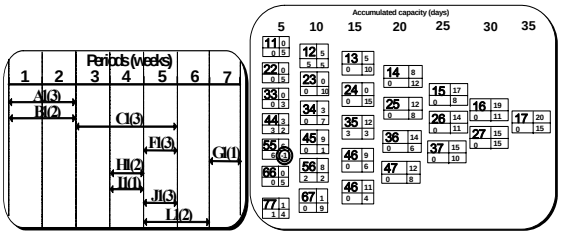


Figure 7: CRP and PFCG of processor P1 after assigning all the available margins.

We check then the feasibility condition. If for a lapse [a,b] in the processor j, the validation condition is not verified, we should lengthen the scheduling time segment of the tasks as follows:

- We consider in the duration increasing order the list of all the tasks included in the lapse [a,b] and that b_{ij} = a or e_{ij}= b.
- In order to treat the minimum number of tasks, we begin to treat in this list the first task in the k^{th} position in the list which has a load equal or superior to the overloading in the lapse.
- We try to regain a margin for this task by shortening the scheduling time segment of the precedent task of the same job in a processor already treated.
- We try this for all the tasks positioned in the $(k+1)^{th}$ position in the list until succeeding or arriving to the end of the list.
- If we arrive to the end of the list, we try with tasks positioned in the $(k-1)^{th}$, $(k-2)^{th}$ … position until succeeding or arriving to the beginning of the list.
- We must lengthen the scheduling time segment of as many tasks as necessary to validate the feasibility of the problematic lapse.
- If the remaining capacity continues to be negative, we reiterate the treatment of the tasks in the same order but by trying in this case to move completely if possible the precedent task of the same job in the past which allows us to lengthen the scheduling time segment of tasks of this processor.
- If we do not succeed, we treat the tasks in the same order by trying to delay the due date on a minimum number of jobs or by increasing the capacity of the incriminate lapses.

If we take our example, the control feasibility graph of the processor P1 indicates that we have a problem in the lapse [5,5] (Figure 7): the accumulated capacity requirement is 6 days and the available capacity is 5 days, so we should lengthen the scheduling time segment of one of the tasks F1

or J1. If we lengthen the task in the right, the due date order will be delayed (F1 or J1 are the last task of the jobs F and J), so we will try to shorten or move the scheduling time of F2 or J2 in the processor P2. F2 can not be shortened (the scheduling time segment is one period). J2 can be shortened, the scheduling time segment will be shortened to the lapse [2,3]. We verify that the control feasibility graph of P2 remains valid. The scheduling time segment of J1 will become [4,5]. The control feasibility graph of P1 becomes valid and we can try to regain margins like done for the processor P2 in the fourth step. We obtain the C.R.P. of processor 1 as follow:

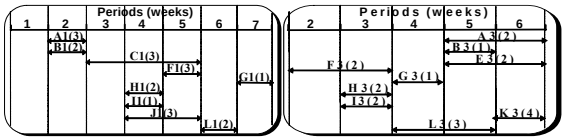


Figure 8: C.R.P. of processor P1 and P3 after treatment.

Then, we apply step 5 for the last processor P3 and we obtain the C.R.P. as shown in Figure 8. We update the beginning and the ending of the scheduling time segments of the other tasks on the other processors as explained in step 4 and we obtain the final CRP of processors P1 and P2:

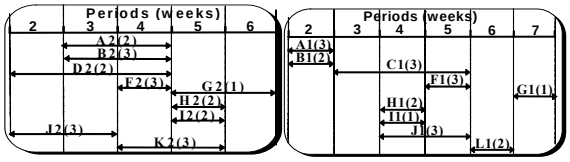


Figure 9: Final C.R.P. of processor P2 after treatment and a possible scheduling solution.

4 SCHEDULING

In case where we admit to interrupt at least one task by period, the overlapping load adjustment approach furnishes a necessary and sufficient condition for the existence of a feasible scheduling solution. The scheduling will be made in a real time manner. Indeed, in the end of each task the responsible will choose the next task between all the tasks that can be loaded and so on. This load adjustment approach will be coupled with a scheduling tool which can function as described below. If we take our example of Figure 1, the real time scheduling can be made as follow:

- In the beginning of week 2 we can choose to begin the task F or B (Figure 1), we choose for instance task B and then task F.
- We arrive at the end of day 1 of week 3 and we see that we can choose between three tasks: A or D or E (Figure 1). The scheduling tool will inform the user if he could really choose one task and during how much time without breaking the feasibility condition. If the user chooses for example task A, the scheduling tool will tell him that the task A can be scheduled for only one day until the end of the second day of the week because there are two tasks E and D with a total duration of 3 days that must be scheduled on week 3. So, if the user decides to choose A, he must interrupt A after one day, schedule E and D and after that, continue with the task A, but the user can choose to schedule E and D without interruption. We suppose that the user chose the task E and so we obtain the partial scheduling described in Figure 10.
- We continue in the same way the scheduling and we can obtain for example the final scheduling of Figure 10.

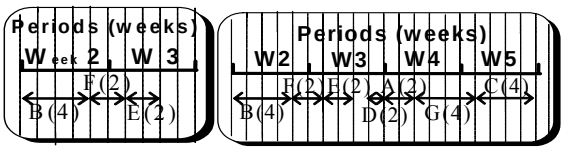


Figure 10: the partial and final scheduling.

In our opinion, this original approach of scheduling presents many advantages in comparison with the automatic calculation of a schedule. First, the approach is really dynamic, each decision is taken in the last moment and we do not produce plans which will be out of date. Moreover, the user can have its own reasons to choose one task or another. A system that proposes and does not impose but exposes the consequences of each choice in regard to the scheduling which allows the user to decide with full knowledge of the facts and integrates his own criteria. This characteristic could promote the scheduling performance, and allow in all cases the responsibility of the user and his comprehension of the system.

5 EXPERIMENTATION

An experimentation has been carried out on a set of examples. We'll describe very briefly (due to edition constraints) the conditions and the results of this

experimentation. We have constructed 20 examples where we varied the number of jobs (5 to 20) and the number of processors (2 to 4). The total load is always near or superior to the capacity. We compared our approach with the classic approach of placement. For each example, we apply our approach and the placement approach. This placement is applied in two steps: first an earliest placement to determine the earliest due dates and second a latest placement using job's due dates as the maximum between the requested due dates and the earliest ones. We have compared the two approaches in terms of number of jobs delayed. The percentage of jobs delayed with the placement is about 25 % whereas it is equal to 8 % with our approach. This fact proves really the efficiency of our approach. However, we should experiment and compare our approach with other approaches.

6 CONCLUSIONS

We think that the most common approach used in production planning remains MRP II (Manufacturing Resource Planning). The proposed approach in this paper works in a hierarchical production planning and scheduling and constitutes an alternative to the traditional load adjustment approaches used in the CRP (Capacity Requirement Planning) modules in software based on MRP II philosophy. The new heuristic presented in this paper, in comparison with the usual middle and/or long-term planning and scheduling approaches, has the following advantages:

- not setting a long-term tasks scheduling to assure that the planning can be properly carried out;
- exploiting the intrinsic margins of each job to obtain their loading time segments guaranteeing the production planning feasibility under the assumption of pre-emptive tasks;
- distributing judiciously the job's margins on their tasks and trying to respect the just-in-time principles;
- splitting up the production planning into jobs subsets making thus its analysis and its exploitation easier;
- permitting the postponement of the final scheduling jobs problem until the short term at the order release phase and/or the scheduling phase;
- delaying, if necessary, the due dates of some jobs or increasing the capacity in some lapses for guaranteeing in every case the feasibility of the production planning.

We can extend and improve our work by studying the possibility of introducing the overlapping of the scheduling time segments of consecutive tasks. We can also improve our heuristic accordingly since we want to minimize the average tardiness or the max tardiness or the number of delayed jobs.

REFERENCES

Bahroun, Z., Jebali, D., Baptiste, P., Campagne, J-P. and Moalla, M., 1999. Extension of the overlapping production planning and application for the cyclic delivery context, in *IEPM '99 Industrial Engineering and Production Management*, Glasgow.

Bahroun, Z., Campagne, J-P. and Moalla, M., 2000a. The overlapping production planning: A new approach of a limited capacity management. *International Journal of Production Economics*, 64, 21-36.

Bahroun, Z., Campagne, J-P. and Moalla, M., 2000b. Une nouvelle approche de planification à capacité finie pour les ateliers flow-shop. *Journal Européen des Systèmes Automatisés*, 5, 567-598.

Dauzere-Peres, S., and Lassere, J-B., On the importance of scheduling decisions in production planning and scheduling. *International Transactions in Operational Research*, 9(6), 779-793.

Dillenseger, F., 1993. Conception d'un système de planification à moyen terme pour fabrications à la commande. PhD thesis, INSA Lyon, France.

Drexl, A. and Kimms, A., 1997. Lot sizing and scheduling - Survey and extensions. *European Journal of Operation Research*, 99, 221-235.

Fleischmann, B. and Meyr, H., 1997. The general lot sizing and scheduling problem. *Operation Research Spektrum*, 19(1), 11-21.

Gunther, H.O., 1987. Planning lot sizes and capacity requirements in a single stage production system. *European Journal of Operational Research*, 31(1), 223-231.

Hillion, H. and Proth, J-M., 1994. Finite capacity flow control in a multi-stage/multi-product environment. *International Journal of production Research*, 32(5), 1119-1136.

Néron, E., Baptiste, P. and Gupta, J.N.D., 2001. Solving Hybrid Flow Shop problem using energetic reasoning and global operations. *Omega*, 29, 501-511.

A SERVICE-ORIENTED FRAMEWORK FOR MANNED AND UNMANNED SYSTEMS TO SUPPORT NETWORK-CENTRIC OPERATIONS

Norbert Oswald, André Windisch, Stefan Förster
European Aeronautic Defence and Space Company - Military Air Systems
{norbert.oswald, andre.windisch, stefan.foerster}@eads.com

Herwig Moser
University of Stuttgart - IPVS
herwig.moser@ipvs.uni-stuttgart.de

Toni Reichelt
Chemnitz University of Technology
toni.reichelt@informatik.tu-chemnitz.de

Keywords: Network-centric Operations, Autonomous Systems, Service-oriented Architecture.

Abstract: Network-centricity and autonomy are two buzzwords that have found increasing attention since the beginning of this decade in both, the military and civil domain. Although various conceptions exist of which capabilities are required for a system to be considered network-centric or autonomous, there can hardly be found proposals or prototypes that describe concrete transformations for both capabilities into software. The presented paper reviews work accomplished at EADS Military Air Systems driven by the need to develop an infrastructure that supports the realisation of both concepts in software with respect to traditional and modern software engineering principles, e.g., re-use and service-oriented development. This infrastructure is provided in form of a prototypical framework, accompanied by configuration and monitoring tools. Tests in a complex scenario requiring network-centricity and autonomy have shown that a significant technical readiness level can be reached by using the framework for mission software development.

1 INTRODUCTION

Network-centricity is a concept that becomes increasingly interesting to both, the military and civil domain. It represents more than just connectivity across systems and nodes, but between people in the information and cognitive domains (NCOIC, 2005). Be it military operations, international peacekeeping missions, large-scale commercial applications or natural disasters, the complexity requires cooperating entities, collaborating to provide sufficient information, resources and services to the collective. As some mission are considered to be "dull, dirty and dangerous" (Freed et al., 2004), the use of unmanned autonomous systems becomes an increasingly popular alternative. Hence, manned *and* unmanned systems need to be considered within *Network-centric Operations* (*NCOs*).

Prerequisite for a fast, efficient and effective collaboration in highly-dynamic missions is the creation of a common understanding based on distributed situational information by means of an adequate IT infrastructure, comparable to the *Global Information Grid* (*GIG*) (Alberts, 2003). Participating systems in NCOs do not necessarily form a homogeneous collective but are usually composed of a patchwork of disparate technologies, such as different communication protocols, transport medias, software architectures, processing platforms, or knowledge-based systems. The consolidation of these technologies demands a proper architectural concept for system interconnection and the availability of a corresponding framework implementation. This problem is acknowledged by the DoD Architecture Framework (DoD, 2003a) but lacks an implementation.

The construction of such an architectural framework is a very complex task due to its interdisciplinary nature. Although plenty of applicable software standards (CORBA, FIPA, HLA, WSDL, ...) exist, composition is difficult because standards usually were designed independently of each other but in this context have to work together. A further problem

arises from the fact, that it is partly unclear how technologies will evolve. Up to now, only few correlation efforts have been made, e.g., JTA (DoD, 2003b) and NC3TA (NATO, 2005), but those lack software engineering aspects. Projects which do provide an implementation, e.g., OASIS (OASIS, 2006), COE (SEI, 1997) or JAUS (JAUS, 2006), concentrate on particular aspects but lack a holistic generic approach.

Thus, in this paper we combine NCOs and autonomous systems approaches by presenting a framework capable of constructing and assembling mission software sufficing the needs especially of airborne, but also ground-based or maritime participants, either manned or unmanned in collaborative missions.

2 DISTRIBUTED AUTONOMY REFERENCE FRAMEWORK

2.1 Design Principles

The conceptual development of a framework is driven by the need to cover two fundamental principles, that is, autonomy and NCOs. The combination of both principles is modelled in the following using a military analogy, but results can easily be mapped to the civil domain. While NCO serves in both domains to gain information superiority, autonomy in military systems is usually associated with unmanned vehicles but can also appear in command and control structures. A typical example is the so called *Auftragstaktik* (von Clausewitz, 1832). The core question is on how to transform these basic principles thoroughly into software.

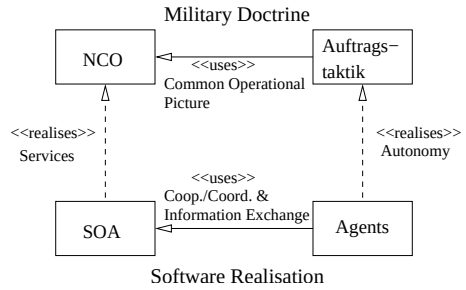


Figure 1: Mapping of military doctrine to realisation in SW.

Figure 1 illustrates our view of the interplay of military concepts transformed into the IT world, and is split into an upper part, the Military Doctrine, and a lower part, the Software Realisation. Auftragstaktik, as shown on the upper-right side of the diagram, is a military doctrine to give a sub-commander a goal to achieve and leave a high degree of freedom concerning actions for goal attainment (this equals "executive autonomy" as defined in (Castelfranchi and Falcone, 2003)). In a dynamic environment, such as a battlefield, Auftragstaktik heightens efficiency through adaptability to changing situations. The principle of Auftragstaktik is also common in the civil domain as exemplified by company hierarchies, where superiors expect their subordinates to work independently but goal-directed. Information exceeding a local view, through the use of network-centric capabilities (i.e., «uses»), enables participants to better decide on how to achieve their goals.

The lower part of the diagram shows the relevant software concepts which we propose to implement the doctrines. The desired autonomy is realised through agent technology, which in turn, is based on and makes use of a *Service-oriented Architecture* (*SOA*) e.g.(W3C, 2004) (SCA, 2006). The use of SOA enables units to become part of the NCO context, by transparently providing and requesting services respectively information to and from other vehicles.

The general approach is thus to provide an infrastructure which enables the creation of a service-oriented architecture, supporting aspects to build systems exhibiting autonomous behaviour. The way *how* services are internally structured and implemented (e.g., in form of an agent) and *where* they are located is transparent to the user.

2.2 Framework Features

We provide an infrastructure called *Distributed Autonomy Reference Framework* (*DARF*) based on the following core features:

Building Blocks: Providing service containers for functionality to ease development of new services or integration of existing code. The DARF includes a pre-fabricated building block, the Service Broker which is described separately.

Situational Assessment: Providing means to build an operational picture composed of shared distributed information, complement the world view and to structure information using ontologies.

Decision Support: Providing means for the integration of reasoning and inference mechanisms, usage of declarative knowledge and to assist the decision making process.

Network Communication: Providing a middleware capable of communication in heterogeneous networking environments and platforms, adaptation of transport media by IPv6 and service directory service.

Simulation Connector: Integrate framework Building Blocks (see Section 2.3) into complex simula-

tion environments using High Level Architecture (IEEE, 2000) standard technology.

This infrastructure forms the basis for the construction of services to be used in network-centric and autonomous scenarios. Furthermore, the framework offers support for the systems engineering process via system configuration, deployment and runtime execution monitoring.

2.3 Building Blocks

The DARF provides service containers which support the construction of new services embedding mission functionality. By turning mission functionality into services, it becomes possible to distribute and use services transparently of their location, and thus independently of which platform[1] provides them.

The DARF provides containers for the implementation of *intelligent* and *primitive* services. This distinction is useful, as some services require decision-making capabilities to fulfil their purpose. The "primitive" qualifier does not imply that the logic behind a service is trivial. It might encapsulate a highly complex legacy functionality like, e.g., automatic target recognition. Hence, primitive services, besides newly developed functionality, may pose as a facade to a legacy subsystem for which a standard way of integration has been developed as described in Section 2.3.4.

Figure 2 shows the meta-model of the building blocks. The abbreviations introduced in the figure are explained in the following sections.

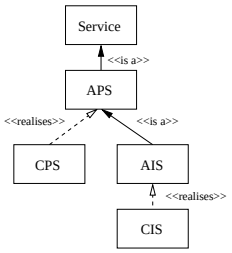


Figure 2: Meta-model of the abstract respectively concrete primitive and intelligent services.

2.3.1 Services

Both, primitive and intelligent services, have in common that they advertise their service description/interface and may request services from others, which first have to be discovered. This baseline functionality is provided by the DARF and the *Service Broker*, a distinctive service registry component, described in Section 2.3.5. Services carry meta-data containing their interface description, configuration and *Quality-of-Service* (*QoS*) parameters (see (Oswald, 2004)). The DARF uses this meta-data for service start-up, registration and monitoring

At runtime, the DARF provides background management functionality which relieves service developers of writing tedious boilerplate code. That is, for registration, the DARF automatically publishes the service description and continues to asynchronously check whether its publication is still valid. This is necessary as the Service Broker may die and become replaced, in which case the service gets re-registered automatically.

The DARF provides client-side proxy classes (i.e., stubs) for every service. Upon the first invocation, respectively after having lost connection to a service, the proxy asks the Service Broker for a list of services ("lazy resolution") having a certain type and sufficing a number of criteria prescribed by the client. The returned list of services is then further sorted according to a proprietary metric and results in the actual invocation on the most appropriate service. This decision-theoretic selection process is described in detail in (Oswald, 2004).

2.3.2 Primitive Service

As mentioned before, services come in two flavours. The first, and more basic one, being of the type *Abstract Primitive Service* (*APS*). When adding functionality and thus *implementing* a primitive service, it becomes a *Concrete Primitive Service* (*CPS*). A CPS is a leaf node component in a hierarchy of services, representing a concrete atomic self-contained functionality not explicitly requiring cognitive abilities. A CPS may be a facade hiding a legacy subsystem.

2.3.3 Intelligent Service

The second type of service is the *Abstract Intelligent Service* (*AIS*), having two main characteristics. First of all, they may make use of other services and second of all, the decision on which services to use respectively what actions to take is not hard-coded but determined intelligently. Note that we agree with (Clough, 2002) in that intelligence is not the same as autonomy but will nevertheless equate intelligent services with autonomous entities. Thus, similar to the Auftragstaktik, intelligent services are given a goal and are free (and capable) to choose how to attain it. The DARF provides facilities for the implementation of rational agents. Related work in this field can be found in (Karim and Heinze, 2005), employing Boyd's *Observe Orient Decide Act* (*OODA*) reasoning cycle, the accepted model of cognition of aircraft

[1] In the military domain, the term "platform" is typically correlated with a vehicle.

fighter pilots. The OODA loop is also the basis for the operation of intelligent services and can be based on a *Belief Desire Intention* (*BDI*) (Georgeff et al., 1999) model as they are, in principle, quite similar.

Just as before, an *Abstract Intelligent Service* (*AIS*) becomes a *Concrete Intelligent Service* (*CIS*) when implemented.

2.3.4 Legacy Wrapper

Development of and for the DARF has to take into account that there exists a large code base of proven-to-work potentially certified code. It can not be assumed that such code is in an easily accessible state. That is, legacy code needs to be seen as something almost entirely opaque. Still, a way of integrating this functionality into the SOA needs to be provided as not all functionality can simply be redeveloped to fit the DARF interface. To tackle this problem, we propose a standardised way of wrapping functionality by using CORBA. Observe Figure 3, illustrating on how this is done. Some legacy functionality $f(x)$ written in language l should be included. For that, it is wrapped/adapted and made available by promoting the wrapper's interface as a CORBA server having an OMG IDL interface. The choice of using CORBA stems from the fact that there exist implementations for almost any programming language and platform available. Note that the *Wrapper Server* for $f(x)$ inherits its operations from the corresponding CPS. This structural pattern is similar to the *Wrapper Facade* described in (Schmidt, 1999) and shares elements with the *Adaptor pattern* from (Gamma et al., 1995). The Wrapper Server is managed by the *Wrapper Manager*. The manager is partly provided by the DARF in form of an abstract super-class, capable of starting up the Wrapper Server and retrieving its CORBA IOR (see (Siegel, 2000)) to instantiate a client stub. The stub is used by the CPS to access the legacy functionality via remote method invocation.

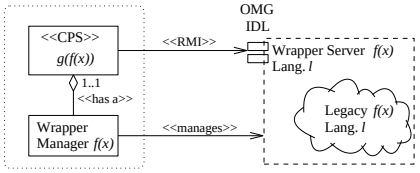


Figure 3: Wrapping legacy functionality $f(x)$ in language l using the DARF wrapper concept.

2.3.5 Service Broker

The *Service Broker* (*SB*) is a centralised service registry which itself is implemented as a service. It is known to every service by providing its persistent reference in every service's meta-data. The SB collects service descriptions from services joining the system and provides look-up facilities to service enquiries. More than that, the SB constantly checks the consistency of its references to services and discards broken ones. Also, as each service may be replicated arbitrarily, it promotes service replicas to service masters (i.e., the only active replica of a service) upon detection of a broken reference to the current master.

It should be noted that the SB is strictly limited to service registration and discovery (it should thus not be confused with the *Broker pattern* in (Buschmann et al., 1996)).

2.4 Situational Assessment

A Blackboard as described in (Craig, 1995) is the enabling technology to achieve an operational picture which forms the basis for any kind of potential actions and reactions of a single platform. The DARF provides a distributed version of a Blackboard above a DDS like system (OMG, 2005) to collect and exchange information. From the software architecture point of view a Blackboard is a service used to share information and knowledge between other services. Both types of services might act on the one hand as sources to provide processed data and inferred knowledge to others by publishing it on the Blackboard. On the other hand, services might subscribe for particular information on the Blackboard.

As with service descriptions and naming, one has to make sure during the engineering process, that all represented information on the Blackboard will be suitable for exchange between platforms. The use of an agreed ontology guarantees, that a common language is used. This is a key element in network-centric operations as usually platforms of different vendors with different systems are participating in missions. The DARF intends to support embedding and usage of various ontologies in the framework to better structure the information, which requires national and international agreements, currently being under investigation.

2.5 Decision Support

Decision making models like Boyd's well known OODA loop (Karim and Heinze, 2005) have been widely used as the basis for developing Decision Support systems. They are tightly coupled with situational assessment and require, apart from their implementation, a number of infrastructural measurements. The DARF supports the embedding of Decision Support fundamentals with a number of features. Firstly, DARF provides the internal structures of the

AIS building blocks to directly support the OODA phases, including access to the distributed Blackboard for information and knowledge exchange. Secondly, DARF provides means to embed existing reasoning and inference methods based on the legacy wrapper concept. This includes handling of varying representational structures, requiring adaptation to comply with the one internally used for situational assessment. Information and knowledge are explicitly accessible in a declarative manner. The declaratively described rules can be changed interactively during runtime execution, by the human system operator or possibly by a CIS possessing learning algorithms.

The developed mechanism for Decision Support assists the platform in a highly dynamic environment by suggesting appropriate actions and while considering human interaction.

2.6 Network Communication

Network communication is based on IPv6 to hide transport media specifics (Reichelt, 2006). IPv6 comes with a number of built-in capabilities which relieve the DARF from having to take care of them. For instance, IPv6 offers encryption, authentication and auto-configuration of network nodes. Using IP(v6) allows the use of various standard IP-enabled technologies. This allows for easy replacement or extension of higher layer protocols or applications to increase the flexibility of the DARF. Driven by the US *Department of Defense* (*DoD*), essential technology standards for NCOs and inter-system connectivity, i.e., the *Global Information Grid* (*GIG*) (Alberts, 2003) and the *Joint Tactical Radio System* (*JTRS*) (North et al., 2006), are all based on IP(v6). The DARF seeks DoD compliance in that respect.

The current prototypical implementation of the DARF is written in Java and based on a CORBA middleware (Siegel, 2000) for inter-service communication. The decision to base the communication on CORBA originates from its wide support for heterogeneous distributed computing but binding to the middleware is kept as loose as possible.

2.7 Simulation Connector

The frameworks capabilities for connecting possibly distributed simulations is handled by an *High Level Architecture* (*HLA*) connector. HLA is a middleware standard supporting the development of distributed simulations (IEEE, 2000). The HLA's runtime implicitly handles data transport across network connections as well as time synchronisation within a composition of separate simulations.

Within the DARF, HLA is used to validate system functionality for different environments which could not be provided by the experimental environment, e.g., the integration of a flight dynamic model. Furthermore, a HLA model can be used to investigate network or protocol behaviour by using communication simulators to emulate real world system interconnection.

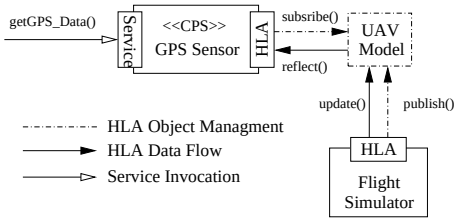


Figure 4: Integration of an HLA-enabled Flight Simulator to provide GPS data via a service interface.

HLA interaction is realised through a CPS making use of a simplified HLA API developed as part of the DARF. Such a CPS provides two different interfaces: A service interface to the framework allowing access to its data and an HLA interface to retrieve the necessary information out of an HLA compatible simulation. Thus, the integration of simulators becomes fully transparent by hiding the HLA interface behind the framework's SOA endpoint (see Fig. 4). This allows easy replacement of the simulated data source by a real-life one (and its appropriate CPS) without any modifications of other system components.

3 SYSTEM LIFE CYCLE SUPPORT

3.1 System Design Approach

Building industrial strength NCO-capable systems with inherent autonomy requires besides the provision of an adequate runtime framework a well-defined systems engineering process and an adequate tool support for mission-specific system configuration, deployment, and runtime-monitoring.

The system design approach used for DARF Building Blocks follows the methodology defined by the *DoD Architectural Framework* (*DoDAF*) (Alberts, 2003; DoD, 2003a). Initially the *Concept of Operations* (*CONOPS*) is modelled which captures the intended operational scenarios of the platform to be built and deployed. This model also reveals the other involved platforms, their operational roles, and the information flows between all of them. Based on this in-

formation, operational requirements for the platform under consideration can be derived and the mission-specific platform functions required for implementing these requirements can be identified. The result of this design step is a function graph, the *systems view*, of the platform. It comprises a set of abstract DARF Building Blocks linked by their service dependencies. In case concrete implementations for the required abstract Building Blocks already exist they can be used immediately. Otherwise the corresponding concrete Building Block software has to be designed, implemented, tested, and integrated into the service pool.

3.1.1 System Configuration and Deployment

The *System Configuration Tool* (*SCT*) assists system developers at design time in composing distributed systems from a given set of existing Building Blocks and framework components. The application of this tool at the system integration stage allows the mission-specific configuration of platforms whilst enforcing both functional (e.g. functional completeness) and non-functional (e.g. Worst Case Execution Times or QoS aspects) system requirements as explained in (Windisch et al., 2002).

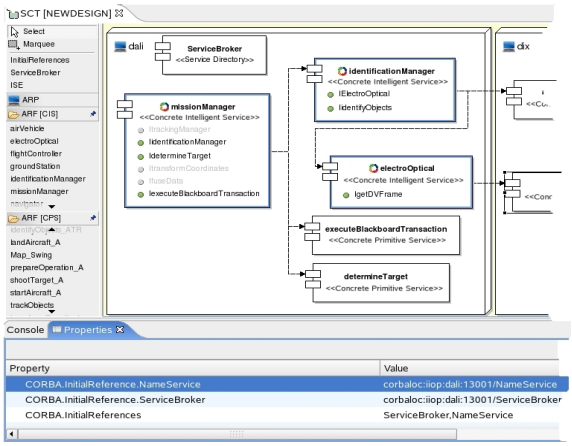


Figure 5: View of the System Configuration Tool.

The SCT, as depicted in Figure 5, is an Eclipse based Integrated Development Environment that comes with a graphical editor for composing distributed systems from a set of Building Blocks. All available CIS and CPS Building Blocks are inferred from the DARF upon tool start-up and displayed by the graphical editor. System integrators can drag and drop required Building Blocks to compose a system.

During this work the tool automatically resolves all service dependencies of the currently used Building Block and graphically indicates any missing services. Service meta-data can be changed by means of the contained property table editor. Once service composition is completed, i.e., a valid system configuration in terms of fulfilled functional and non-functional system requirements has been found, the SCT tool generates all necessary DARF start-up scripts and distributes them across the involved processing nodes.

3.1.2 System Runtime Monitoring

The Mission Monitor tool offers views on both the world scenario and the internal state of all involved platforms. In a sense, the latter view constitutes the runtime counterpart to the SCT tool, as it gains insight into the mission-specific platform configuration including all currently active services. The world scenario view, on the contrary, depicts all involved platforms from the system-of-systems level and provides graphical information on their current state of interconnectivity in the highly-dynamic mission network and currently active cooperations between platforms.

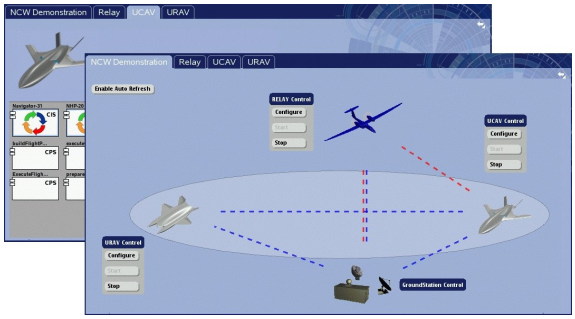


Figure 6: Mission Monitor.

The Mission Monitor is a web-server based extension of the DARF which continuously infers information on service execution and interconnection from the Service Broker components of the involved platforms. This information is subsequently transformed and published by a set of HTML pages which are dynamically created using Java Server Pages. One advantage of this approach is the fact that all information is published in a network transparent manner and, hence, can be obtained by any computing device running a standard web browser.

4 EXPERIMENTAL PLATFORM

To validate our approach, we have selected an instance of a *sensor-to-shooter* scenario which is, in its general case, well understood but still poses a challenge (Chapman, 1997).

4.1 Scenario

The basic idea is the cooperation between two kinds of unmanned air vehicles, one acting as a reconnaissance unit (URAV[2]), essentially being the "sensor", and the other one acting as a combat unit (UCAV[3]), representing the "shooter."

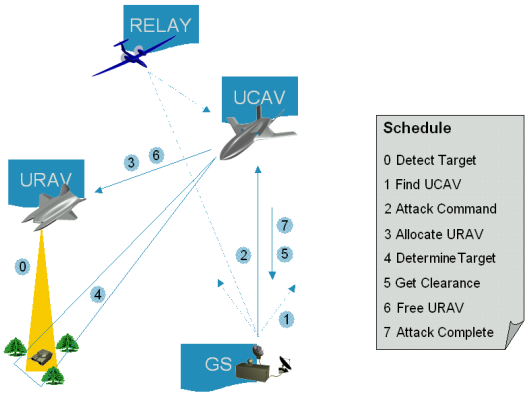


Figure 7: The sensor-to-shooter scenario.

Observe Figure 7, depicting the interaction of the various platforms. Initially, only the URAV is in the mission area, scanning it for potential threats. Upon detection of a target (Step 0) by the URAV, the *Ground Station* (*GS*)'s operator, who monitors the URAV's mission progress, locates an idle UCAV (Step 1) and issues an attack command (Step 2). This commands the UCAV to proceed to the mission area, where it makes contact with the URAV (Step 3) to acquire target information (Step 4). Having received this information, the UCAV is ready to shoot but has to wait for clearance from the GS operator (Step 5). Once the clearance has been granted and the threat was neutralised, the URAV may proceed with its reconnaissance mission (Step 6). The UCAV's mission is complete once it informs the GS of the (successful) outcome (Step 7). Each inter-platform communication within the scenario can potentially be routed through a Relay to increase the communication range.

4.2 System Modelling and Execution

This section shows how DARF is used for system development and mission execution for the scenario described above. Also, it is demonstrated how the DARF supports system validation through simulation.

[2] *Unmanned Reconnaissance Air Vehicle.*
[3] *Unmanned Combat Air Vehicle.*

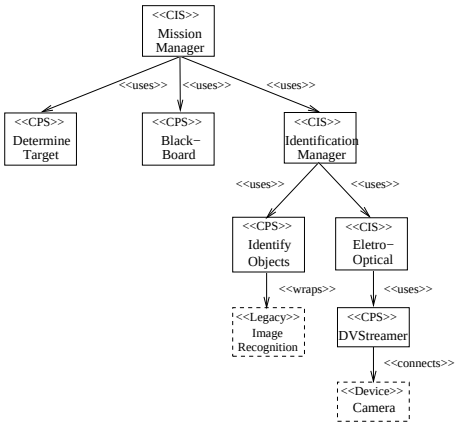


Figure 8: Service Model of target determination.

According to the SOA principles, the desired platform behaviour is broken down into corresponding services. Services are either re-used from a custom pool of existing mission functionality or has to be newly developed by using the approach described in Section 3.1. Figure 8 shows a concrete service decomposition as part of the mission software. One of the *Mission Manager* (*MM*)'s purposes is to acquire and assess sensor information to determine possible target candidates. For this, it makes use of the *Identification Manager* (*IM*) which delivers target classifications. Based on the IM's classification, the MM uses the Determine Target service to select one target out of a potential number of candidates. This target is published on the *Black Board* (*BB*), promoting it to system wide knowledge.

Services use the BB to implicitly distribute knowledge which builds the operational picture of a single platform. BBs can be synchronised between different platform, propagating knowledge from one platform to the other. This corresponds to Step 3 and 4 in our scenario, target information is transmitted from the URAV to the UCAV. The actual target determination is performed by the DARF's reasoning engine using inference rules. Due to the declarative nature of such rules, target prioritisation can be changed during runtime.

As testing would be too risky and costly when performed on real hardware (i.e., real UAVs), simulation has to take its place. Flight controllers of the UAVs interact with simulated flight dynamics models using the DARF's HLA connectivity. Using the same approach, other mission elements, such as radio links, can be simulated to investigate system performance.

5 CONCLUSIONS

Future mission software will have to possess network-centric capabilities. Also, the advent of autonomous systems changes the way missions are executed which in turn influences the way software has to be designed an developed. Complex operations relying on the cooperation of manned and unmanned, possibly autonomous, entities require the possibility of interaction by sharing tasks and exchanging information and knowledge. We developed a framework, the DARF, that uses a combination of SOA and agent technologies to cope with the requirements of NCO-enabledness and autonomy capabilities. That is, each mission functionality is modelled as a service, accessible regardless of platform boundaries. Autonomy related functionality is supported by providing infrastructural means for reasoning and knowledge representation. Additionally, the standardised way of legacy functionality integration allows easy reuse and incremental adoption of the DARF.

To support developers, the DARF comes with a defined engineering process and development as well as maintenance tools. To investigate system performance, the DARF offers a standardised and flexible API to access distributed simulations.

We have validated our approach and shown the applicability of DARF using the well known sensor-to-shooter scenario, exhibiting autonomous cooperative behaviour. Future research on DARF will deal with a unified middleware-independent messaging mechanism, introduction of a general ontology and further extension of the agent-related infrastructure.

REFERENCES

Alberts, D. S. (2003). Network centric warfare: curent status and way ahead. *Defence Science*, 8(3).

Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P., and Stal, M. (1996). *Pattern-Oriented Software Architecture: A Systen of Patterns*, volume 1. Wiley.

Castelfranchi, C. and Falcone, R. (2003). From automaticity to autonomy: The frontier of artificial agents. In Hexmoor, H., editor, *Agent Autonomy*, Multiagent Systems, Artificial Societies, and Simulated Organizations, chapter 6. Kluwer.

Chapman, W. G. (1997). Organizational concepts for sensor-to-shooter world. Master's thesis, School of Advanced Airpower Studies, Alabama.

Clough, B. (2002). Metrics, Schmetrics! How The Heck Do You Determine A UAV's Autonomy Anyway? *NIST Special Publication*, (990):313–319.

Craig, I. (1995). *Blackboard Systems*. Ablex Publ.Corp.

DoD (2003a). *DoD Architecture Framework Version 1.0*. DoD Architecture Framework Group.

DoD (2003b). Joint technical architecture. `http://www.acq.osd.mil/osjtf/pdf/jta-vol-I.pdf`.

Freed, M., Harris, R., and Shafto, M. G. (2004). Humans vs. Autonomous Control of UAV Surveillance. In *1st Intelligent Systems Tech. Conf.*, Chicago, USA.

Gamma, E., Helm, R., Johnsen, R., and Vlissides, J. (1995). *Design Patterns*. Addison-Wesley.

Georgeff, M., Pell, B., Pollack, M., Tambe, M., and Wooldridge, M. (1999). The belief-desire-intention model of agency. In Müller, J., Singh, M. P., and Rao, A. S., editors, *ATAL '98*, volume 1555, pages 1–10. Springer.

IEEE (2000). *IEEE Std 1516-2000: IEEE Standard for Modeling and Simulation High Level Architecture*. IEEE, New York, NY, USA.

JAUS (2006). Joint architecture for unmanned systems. `http://www.jauswg.org`.

Karim, S. and Heinze, C. (2005). Experiences with the design and implementation of an agent-based autonomous uav controller. In *AAMAS '05*, pages 19–26, New York, NY, USA. ACM Press.

NATO (2005). C3 technical architecture. `http://194.7.80.153/website/home.asp`.

NCOIC (2005). An introduction to the network centric operations industry consortium (ncoic). Technical report, NCOIC.

North, R., Browne, N., and Schiavone, L. (2006). Joint tactical radio system - connecting the GIG to the tactical edge. In *MILCOM'06*.

OASIS (2006). Reference model for service-oriented architectures. `http://doc.oasis-open.org/soa-rm/v1.0/soa-rm.pdf`.

OMG (2005). Data distribution portal. `http://portals.omg.org/dds`.

Oswald, N. (2004). Towards a Conceptual Framework-Based Architecture for Unmanned Systems. In *ICINCO '04*, pages 118–126, Setúbal, Portugal.

Reichelt, T. (2006). Communication mapping for enterprise service architectures over heterogeneous transport media. Master's thesis, Chemnitz University.

SCA (2006). Service component architecture. `http://osoa.org/display/Main/Home`.

Schmidt, D. C. (1999). Wrapper facade: A structural pattern for encapsulating functions with classes. *C++ Report*.

SEI, C. (1997). Defense information infrastructure common operating environment. `http://www.sei.cmu.edu/str/descriptions/diicoe_body.html`.

Siegel, J. (2000). *CORBA 3: Fundamentals and Programming Second Edition*. OMG Press.

von Clausewitz, C. (1832). *Vom Kriege*. Dümmlers Verlag.

W3C (2004). Web services architecture. Technical report.

Windisch, A., Fischer, M., and Förster, S. (2002). A re-use orientated design methodology for future integrated modular avionics systems. In *FAST'02*, pages 147–153, London, UK.

A MULTI AGENT CONTROLLER FOR A MOBILE ARM MANIPULATOR

Sébatien Delarue, Philippe Hoppenot and Etienne Colle
IBISC, CNRS FRE 2873 - University of Evry, 40 rue du Pelvoux 91020 Evry cedex, France
sebastien.delarue@ibisc.univ-evry.fr, philippe.hoppenot@ibisc.univ-evry.fr, etienne.colle@ibisc.univ-evry.fr

Keywords: Multi-agent, arm manipulator, mobile platform.

Abstract: Assistive robotics especially mobile arm manipulator can be useful for restoring manipulation function of disabled people in everyday life tasks. However, those systems must be designed to be reliable, fault tolerant and easy to control. This article proposes a method based on multiagent system for controlling the robot. This kind of distributed architecture makes possible to be fault-tolerant without specifying additional management of faults what improves reliability. Moreover, it is also possible to add specific constraints, for example human like behaviors in order to facilitate the use of the system by the person. The multiagent method is easier to implement than classical robotics approaches.

1 INTRODUCTION

ARPH project (French acronym for Robotic Assistance to disabled People), developed in IBISC laboratory, deals with restoring handling function for handicapped person. It is a semi autonomous mobile manipulator arm. Three types of control mode are developed. In automatic one, the operator only gives the goal and the system achieves it automatically. However user would like to do by himself so in manual mode, the operator controls all the degrees of freedom of the system but the user's workload often is too important. The idea consists in developing shared modes in which operator and system share robot control. Project is divided into two research axes: robotics for building autonomous functions and human machine co-operation. The paper focuses on robot control while keeping in mind specificities due to the co-operation between human and machine. For example industrial robot performances such as accuracy are less necessary than easiness of control by the user and fault tolerance, the person being strongly tributary of the assistance.

The classical method to control a robot is to compute mathematical static and/or dynamic models of the robot (Yoshikawa, 1990). The approach provides good results in known environments for carrying out repetitive tasks. If the objective is known in Cartesian space ($p(x,y,z)^T$), those models provide speed or angular value of arm joints so that end effector performs the task. Generally models are computed off-line so they are not modifiable and cannot be adapted easily to quick changes of robot structure e.g. due to the dysfunction of one of robot joints. It requires addition of specific management for making the system fault-tolerant and taking into account on-line changes. In assistive robotics, it is an important criterion which can be solved by exploiting the system redundancy.

In literature, many works exploit redundancy for other objectives, for example to keep an optimal manipulability of the end effector (Yoshikawa, 1990), (Bayle, 2001). Some authors, (Chabane, 2005) (Yoshikawa, 1984), have proposed manipulability measures related to the task to be achieved. Our goal is to propose an only model which exploits robot redundancy and able to perform task even in case of joint default.

In addition, the model must take into account some aspects of human machine cooperation. For instance previous works demonstrated the interest of giving robot human like behaviors for facilitating the appropriation of the robot by the user (Rybarczyk, 2002) (Fuchs, 2001).

Distributed artificial intelligence makes complex problem resolution possible more easily than classical approaches. Our approach is based on a multi-agent architecture divided into two parts, a set of agents for the arm manipulator and only one for the mobile platform. After a bibliographical study, we present the multi-agent architecture able to

control the arm manipulator. Then, the architecture is extended in order to control the mobile arm.

2 AGENTS AND MAS

Computer science evolves from a centralized architecture (sequential treatments) to a distributed one (parallel treatments). So, very quickly are appeared autonomous agents, able to achieve individual task with no external help. MAS for Multi Agents Systems try to solve complex problem which cannot be solved by a unique limited-mind entity (Ferber, 1995). An agent can be defined as a autonomous and flexible software entity (Weiss, 1999). An agent must be able to answer environment changes. An agent has to perceive its environment, to treat the data and then to act. This is a sensori-motor loop. Stimuli may come from the inside (agent itself) or the outside (environment). Action is performed on the agent (internal states) or on the environment. Agent behavior is the result of interaction between the agent and its environment.

It exists three ways to implement agent behaviors: cognitive, reactive, hybrid. Cognitive way divides the internal treatment into three parts perception, planning and action. Agent must have its own world knowledge. It is able to analyze the situation, to anticipate and then to plan an action. Issues of this approach are limited speed and limited flexibility. It is also inadequate in the case of unexpected events. Reactive agent locally perceives its environment (and possibly its internal states) and deduces immediately actions to be carried out only from this source of information. This principle is based on reflex action ((Zapata, 1992), (Wooldridge, 1999)). Hybrid approach merges the first two agents a basic reactive behavior with a high cognitive level. It aims at joining reactivity with thinking and organization abilities of cognitive agents ((Brooks, 1986), (Chaib-Draa, 2001)).

The objective of multi-agents systems (MAS) is to bring together a set of agents and to organize them to reach a high level goal. We can find several kinds of MAS. There are reactive systems which bring together agents and then try to get an emergent behaviour that can solve a higher level problem than each agent can do. Another type of MAS appeared with the need of making agents communicate in order to cooperate (Beer, 1998). Parker (Parker, 1999) uses a central machine which supervises messages. In this case, supervisor organizes groups of agents whose competences are different but necessary to succeed.

Reactive MAS are often used to solve problems with the help of limited-mind agent having poor world knowledge and poor action ability. The objective is to find a social emergent behaviour of an agent society able to solve a complex problem. For example, design of ants or bird behaviour uses this type of approach (Drogoul, 1993). Higher level MAS are frequently used in mobile robotics, especially in collective robotics (Lucidarme, 2003). Systems are based on criteria like gratification, altruism or cooperation (Lucidarme, 2002). A complex dialog is implemented between agents. A high hierarchical level entity is needed to oversee the task to achieve and to centralize decisions and communications. The objective is for example to coordinate several tens of mobile robots transporting containers on quays (Alami, 1998). Some others applications are developed for path planning, including obstacle avoidance (potential fields) associated with artificial life algorithms (Tournassoud, 1992), (Mitul). In robotics an application of MAS is the design of arm manipulator model. We can find few approaches (Duhaut, 1999) (Duhaut, 1993), which describe how to reach a Cartesian position without needing arm inverse kinematic model. The method seems to be well-adapted to our case. Each link is implemented like an agent. Task resolution starts with the link of end effector. It tries to align itself with the goal and to place end effector on the goal by uncoupling itself from previous link. Then, next link does the same with a new sub goal given previously.

Figure 3 shows the beginning of the recursive algorithm applied to a 3 DOF arm. Only three steps are shown in order to illustrate how it works. The goal is the cross. Initial situation is first step. On step 2, the end effector link rotates virtually and uncouples itself from the arm to reach the goal. On step 3, second link rotates and uncouples itself from the arm to join the end effector link. The main characteristic of the algorithm is that end effector link makes bigger rotation than second one and so on. Arm unfolding is not equally distributed between each joint.

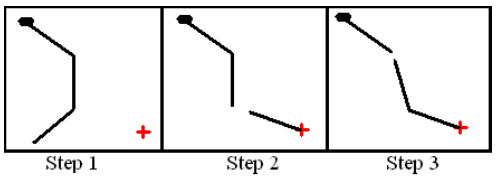


Figure 1: D. Duhaut approach.

Another use of MAS concerns high level complex system like fault detection (Guessoum, 1997) or data merging (vision, touch, sound ...). It often uses neuronal networks or fuzzy rules to find the more pertinent information. In this case the word

"agent" is usually employed in the meaning of "expert system".

3 ALTERNATIVE MAS APPROACH

3.1 Presentation of the Approach

In our approach, we associate an agent to each arm joint. Each agent has the same behavior rules as the others. A joint agent computes the position of the end effector (Figure 2). The objective is to move the end effector as close as possible to the goal. The agent behavior is described by a very simple algorithm:

Agent rotates joint in one direction and computes the new position of the end effector. If this one is closer to the goal than the present one, it is the good direction and rotation continues in that direction. If the new position is not better then rotation follows the opposite direction.

By repeating this algorithm, the end effector moves as close as possible to the goal. In the basic algorithm it is possible that the agent rotates in the bad direction at the first step. We will see in the next paragraph that it can be avoided by using arm direct kinematic model.

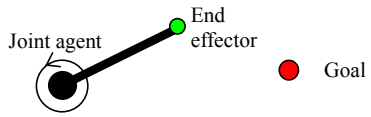


Figure 2: Initial Agent position.

This behavior is now extended to a n-joint arm. Each joint is controlled by an instantiation of the reactive agent described before. The set of arm reactive agents process data in parallel way. Agents are autonomous and each one only tries to minimize the distance between end effector and goal. The algorithm of each joint is implemented in separated processes. As said before we look for an emergent behaviour that satisfies the global mission: the arm end effector must reach the position of the goal.

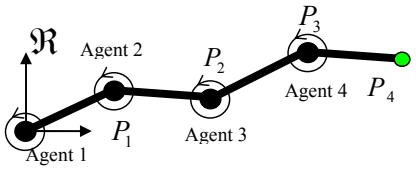


Figure 3: Four agent extent.

Figure 3 shows a four agent example. As previously explained in the one agent case, each agent must know the position of the end effector and the movement generated by its action. It can be done following two ways: first one, all joint values are recorded in real time on a blackboard readable by all agents. So, each joint agent can calculate the position of the end effector by using the simple direct kinematic model of the manipulator. This solution implies to know this model. Second way avoids the knowledge of any model by using an external system (video camera, GPS) that can detect the real position of the end effector. In the first case, as the manipulator model is known, each agent can pre-calculate the end effector position without really moving the joint it controls. So, after a simulation in one direction, it can decide the right move and rotates joint directly in the right direction. In the second case, we don't need models or any joint encoders. We don't need any communication between agents about joint position but we must accept that a joint may rotate in the wrong direction at some steps of the algorithm. What is interesting in this case is that the algorithm can be implemented directly on any manipulator without knowing any arm measurement.

Example of Figure 3 shows the case of a 2D robot. The method is also efficient for a 3D system.

3.2 Results

3.2.1 Comparison with (Duhaut, 1993)

Figure 4 shows the difference between two approaches. The first, on the left, uses our MAS algorithm called MSMA. The second, on the right and called MD, has been developed by Dominique Duhaut (Duhaut, 1993). Simulation presents a five-joint arm manipulator represented on a 2D plane. The arm unfolding is presented in three steps.

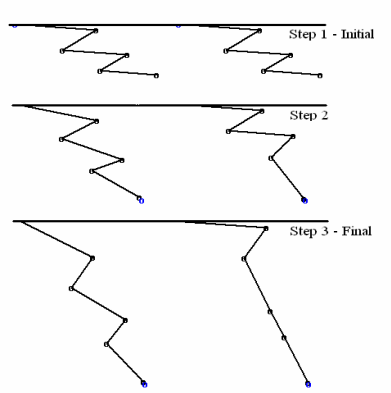


Figure 4: Comparison MSMA and MD.

We can see that during unfolding, the MD approach tries to unfold links starting with the end effector one. The MSMA system unfolds all links at the same time. So, each joint contributes to the movement, avoiding links alignment. Moreover, visual aspect is closer to human behavior. When a person wants to take an object, he does not stretch himself to the maximum.

Figure 5 shows an arm folding. Again, we can see a more distributed movement between joints in the case of the MSMA approach. There is no collision between links.

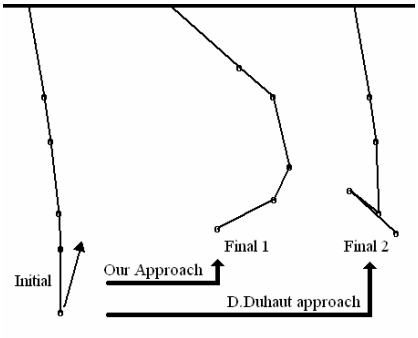


Figure 5: Arm folds.

3.2.2 Behavior in Case of Broken Joint

Duhaut's method (Duhaut, 1993) does not permit to simply take into account a joint fault so we only present results with our method. Figure 6 shows system behavior including two broken joints (dashed limb and squared joint). In this simulation we can use only the end effector position knowledge (no kinematic model, no encoder values). In that case, broken joint means the motor associated to it is out of order (joint encoder out of order or not). If we use a blackboard and the direct kinematic model, as explained in the presentation of the approach, then, the joint encoder must not be out of order to guarantee a correct operation.

Here, the arm works at only 60% of its capacities. Reachable domain is delimited by two half circles. Dotted area represents the reachable space taking into account the reduced capacities. A systematic test has been realized, covering all the reachable space: 100% of the 4592 tested positions have been reached. The figure also shows three sample positions. It is possible because of the redundancy of the system.

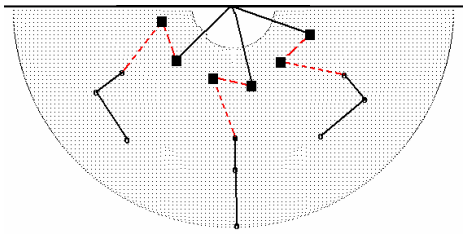


Figure 6: Two broken joints simulation with MSMA.

3.2.3 Discussion

As agents work independently, all joints participate to the movement inducing a natural visual aspect of the arm for the user. For us, this property is very important because it gives the system a human behavior and helps human-machine cooperation. The movement is distributed in all joints and the arm configuration stays far from singular positions. We can also see that there is no link collisions while the arm folds. In the third simulation, without integrating any fault treatment, we can see that MSMA using external localization system is fully fault tolerant if goal is reachable. If a blackboard is used to store joint values in addition to the direct kinematic model, joint encoders must not be out of order even if motor is. In the two cases, even if a motor is randomly moving because of noises or control system fault, the system will reach the goal. We can notice that MAS system does not need to be informed of joints faults. Each software agent does its work without taking care if it really controls or not its associated joint.

4 APPLICATION TO MOBILE ARM MANIPULATOR

The system to control is composed of a manipulator arm embarked on a mobile platform. The first objective deals with human-machine co-operation. The idea is to give to the system behaviors inspired from human being. For example, when a person wants to take a book, he/she tries to keep his/her arm not extended. If the book is too far to be taken by arm extension, the person walks in the direction of the book aligning the body on the hand movement direction. The second objective is to make the system more tolerant to some joint fault by exploiting redundancy without using specifics fault treatments. Indeed, in assistive field, it is important to maintain a good quality of service.

4.1 Mobile Platform Agent

The control of the mobile platform uses an agent which controls angular and linear speeds. The mobile platform agent is more complex than the arm ones. The agent used for the mobile platform is an hybrid one. Its cognitive capacities give the possibility to add some interesting behaviors.

Firstly, the mobile platform has to move forward because sensors for obstacle avoidance are located on the front side of the system.

The second implemented behavior is to align direction of displacement of the mobile platform with the manipulator arm. Indeed, when a person wishes to catch an object and must move to do it, he/she generally tights the arm forward in the direction of the movement of the body.

The third behavior concerns deadlock. In certain cases, both mobile platform and arm shoulder joint rotate at the same speed, but one on the right and the other on the left direction. In that case, the gripper does not converge to the objective. The agent of the platform is able to detect this kind of situation. It introduces a waiting cycle by leaving to the arm the priority for reaching the goal. Once this cycle ends, the platform agent tries again to align itself with the arm if it is possible.

Reactive behavior is the same one as for the arm agents. Mobile platform agent tries to minimize distance between the end effector and the goal. Mobile platform agent works independently from the arm ones.

4.2 Results

We now simulate the whole system algorithm. First we compare it with a classical mathematical method using manipulability constraints. Secondly, we simulate faults on some joints.

4.2.1 System and Protocol

The system is a 3D mobile arm. It is composed of a mobile platform equipped with two driving wheels and a free wheel to ensure stability. A 6 DOF manipulator arm is fixed on the mobile platform. In the following simulations, the end effector imposed task consists in following a straight line with a constant speed, in the upper-right direction of the mobile platform. Figure 7 describes the system and the associated mathematical frame. The displacement is perpendicular to the initial orientation of the platform. The displacement plan is formed by x and y axis. Initial conditions are showed in Table 1.

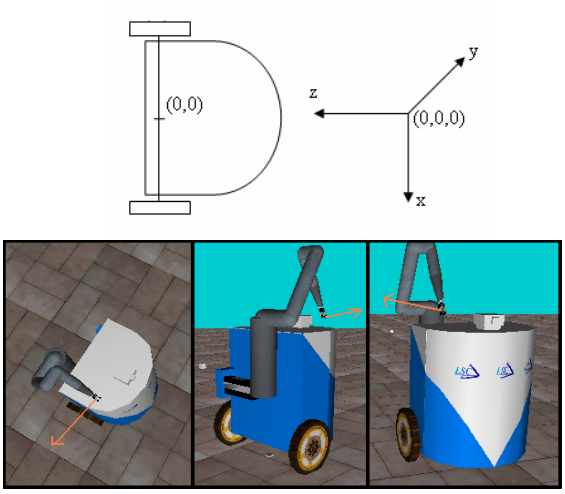


Figure 7: System description.

Table 1: Initial simulation conditions.

Object	Initial value
Platform position	(0,0,15) meters
Manus Joint 1	270 degrees
Manus Joint 2	120 degrees
Manus Joint 3	-125 degrees
End effector position	(18,79,-20) meters
Simulation steps	400
Sampling rate	60 ms
Shift wanted for each step	(0.42 , 0.12 , 0) cm
Total duration	24 s

On the first simulation, objectives are the following ones:

- End effector must follow the desired trajectory with good accuracy
- Mobile platform must move forward because of front side sensors belt
- Mobile platform has to align itself with the arm orientation
- Arm has to avoid its extended configurations.

We compare our approach (MSMA) using agents introduced above with the MIM one which uses manipulability criterion (Chabane, 2005).

Secondly, we check fault tolerance ability of MSMA approach. So, we simulate a breakdown of the shoulder and check if the whole system redundancy (generated by the mobile platform) permits to the system to reach the goal. We also simulate a breakdown of joint 2 to check the whole system behavior.

4.2.2 Results

Her, we do not show end effector trajectory. With both approaches, the move is correct with a good accuracy (less than 3mm of difference between the real path and the desired one). There is no notable

difference between them. Accuracy is linked to simulation step duration. We choose a high one of 60 ms because our real system has a 60 ms command loop.

Figure 8 shows platform trajectory and orientation. We can see two very different performances. MSMA works well. It always goes forward keeping obstacle avoidance sensors in front. It first turns slowly on the right, and then goes straight until the end and aligns itself with the arm orientation. We see a graining point in MIM simulation and the platform ends the task moving backwards.

During the move and with both models, the arm is never bended. Angle between joint 2 and joint 3 stabilizes to a 70 degrees value which is far from the 0 degrees singular value of the arm.

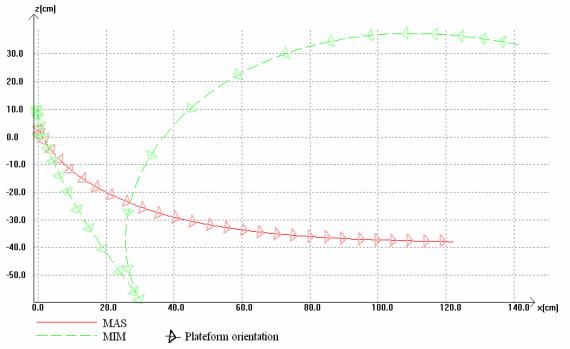


Figure 8: MIM and MSMA platform moves.

We now show how the system reacts in fault cases. MIM is not designed to be fault tolerant so its behavior is not interesting here. It is why next simulations show only MSMA model in three samples faults situations:

- Arm breakdown joint 1 (shoulder) at 60°
- Arm breakdown joint 1 (shoulder) at 30°
- Arm breakdown joint 2 at 120°

These values are chosen as examples to illustrate motor joint breakdown fault tolerance. We assume that if goal is reachable, fault tolerance is effective in all the cases. Breakdown occurs at time 0. Figure 9 shows end effector trajectory on x axis. We can see that MSMA satisfy objective even in fault situations. Results are the same on y and z axis. In the first two situations, redundancy between arm joint 1 and the mobile platform rotating movement is exploited. As end effector follows the wanted path, we can say that the platform agent works well when the arm shoulder is broken. In that case, mobile platform moves forward but does not align with the arm orientation. Indeed, this alignment is performed by the redundancy between arm joint 1 and the mobile platform rotating movement and, in that case, joint 1 is broken. Moreover, the arm is not tightened and the angle between joint 2 and 3 is stabilized to 70 degrees. In the third situation, redundancy between joint 1 and joint 3 is exploited. Both of them permit a vertical move (y axis). Once again the end effector follows the trajectory correctly. In that case, the mobile platform moves forward and aligns with the arm orientation. We can still notice that the arm is not tightened and that its posture remains far from singular position.

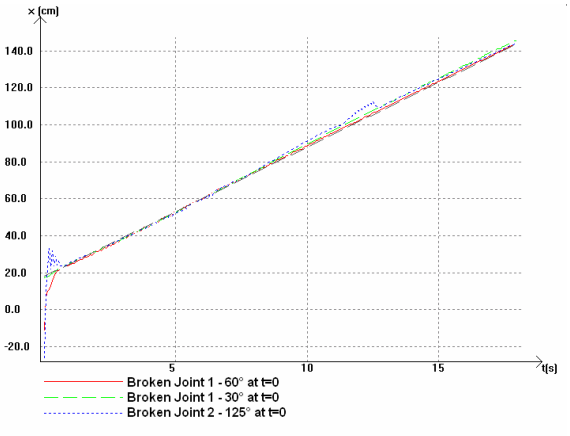


Figure 9: End effector trajectory on axis x.

4.2.3 Discussion

On the first simulation, we can see that the classical model used implies a more complex trajectory for the mobile platform on the whole system simulation. There is a graining point making the robot blind (it can not avoid obstacles anymore because ultrasonic belt is in front side). It's due to the manipulability criterion used in this simulation. Our model shows a better behavior, closer to a human one. The platform goes forward and aligns itself on the arm orientation inducing a more assimilable move for the user. The use of independent agent helps to considerate directly human behavior in algorithm. Also, the system is fault tolerant as shown in simulation. The end effector desired trajectory is reached even if arm joint 1 or 2 are broken. This ability is due to multi agent architecture which is able to run even if a component is defective.

5 DISCUSSION

First, with our approach, we do not cut out the final objective in sub goals which each agent would have to reach. Each one has a local work to do independently from others. We do not organize any co-operation. Here, we speak about emergent

behavior. Indeed, one agent can not reach the goal alone. It needs others to achieve the task but it does not know it. This kind of system provides very good result concerning fault-tolerance.

Secondly, this approach induces a goal for each agent. It is then possible to influence behavior of some agents. Then we can easily adjust or add behaviors to facilitate man machine co-operation. That is the case for example with platform alignment on the direction of the end effector in the same way than alignment of the human body on its hand direction. That leads to an easier assimilation of the system by the user. Indeed, with a classical model, to add a behavior requires the integration of constraints in the global model itself, which is not easy with this kind of system.

Our approach has also its limits. If we integrate many behaviors, it is possible for the system to loose its wanted emergent behavior. The added constraints could be in conflict with the initial objective which is to reach the goal. The system then enters in non convergence cases. At the same time, we can loose fault tolerance ability. To avoid this kind of errors, it is first necessary for agents to keep autonomy compared to the goal they have to achieve and thus to be as simple as possible. Secondly, we have to include priorities in the local objectives of each agent. Reaching the goal has a high priority, going forward has a smaller one. Aligning platform on arm orientation has a very small one. Thirdly, we have to implement deadlocking treatment by introducing delays in specific situations. In our system, these particular treatments are implemented in the agent of the mobile platform. We do not plan any problem resolution between agents. In our approach, we keep simple algorithm to avoid high hierarchical management. Indeed, high hierarchical management could then be compared with a system using mathematical models and including a fault treatment supervisor to switch between them.

6 CONCLUSION AND PERSPECTIVES

Our MAS system gives good results in relation to human behavior. Objects can be caught with an easily assimilable movement for the user (forward move, alignment of the platform with arm orientation, simple trajectory with no turnaround, no bended arm). Accuracy is similar to classical method. It is fault tolerant without integrating any specific treatment. It makes easier the integration of special human driving mode. There is no need to compute mathematical model and especially the inverse model. Our MAS system algorithm is easy to implant and need only some geometric formulas and thus very little computing power. It is a real time one. We now have to implement algorithm on our mobile platform and create scenarios of displacement to judge the relevance of our algorithm on a real robot. The first tests on the real system give encouraging results. Simulations shown in this article give an idea of the mobile manipulator behavior with straight lines trajectory. These kinds of trajectories are usually used by people driving ARPH with HMI actually developed. So we think that MAS system can directly replace the Cartesian internal manipulator mode actually used. Actual HMI allows user to control the platform and the arm manipulator separately (two sets of buttons) as shown on Figure 8.

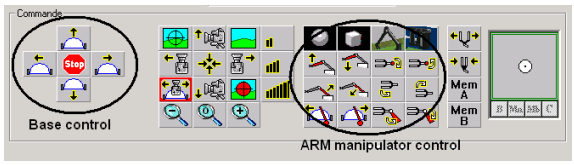


Figure 10: The actual HMI two sets of buttons.

With MAS, it's possible to control the whole system with only one set of buttons by driving only the grip, thus limiting the difficulty of seizure operation by the user.

We also want to improve object grasping. One possible way is to integrate a neuronal network that could help the system to have a better posture (eg: catching an object by the left if user is a left hand writer...). This improvement should lead us to manage with agents not only for the mobile arm but also for the orientation of the grip. Moreover, ARPH mobile platform has an ultrasonic sensors belt for obstacle avoidance when moving in a room that has to be integrated with MAS system to have a fully operational system.

REFERENCES

Yoshikawa, T.: Foundation of robotics: *"Analysis and control"*, The MIT Press, 1990

Bayle, B., Fourquet, J.Y. and Renaud, M.: *"Manipulability Analysis for Mobile Manipulators"*, ICRA'2001, Séoul, South Korea, May 2001, pp. 1251-1256.

Khiar Nait Chabane; Philippe Hoppenot; Etienne Colle, *"Mobile Arm for Disabled People Assistance Manipulability Analysis"*, International Computer Systems and Information Technology Conference, Algeria, July 2005

Yoshikawa, T.: *"Analysis and control of Robot manipulators with redundancy"*, In M. Brady & R.

Paul. Robotics Research: The First International Symposium, MIT Press, 1984, pp 735-747.

Yves Rybarczyk, Etienne Colle, Philippe Hoppenot: *"Contribution of neuroscience to the teleoperation of rehabilitation robot"*, SMC'2002, Hammanet, Tunisia, 6-9 october

Fuchs, J. Moreau, G et Papin, *"Le traité de la réalité virtuelle"*, Nantes : Les Presses de l'Ecole des Mines, 2001

Jacques Ferber, *"Les systèmes Multi agents Vers une intelligence collective"*, IIA Informatique Intelligence Artificielle InterEditions, september 1995

Weiss G., *"A Modern Approach to Distributed Artificial Intelligence"*, The MIT Press, Cambridge, Massachusetts , 1999

R. Zapata, P. Lepinay, C. Novalés and P. Déplanques, *"Reactive behaviors of fast mobile robots in unstructured environments : sensored based control and neural networks."* SAB, December 1992.

Wooldridge M., Weiss G., "Intelligent Agents" *Multiagent systems*, - Ed.: MIT Press - 1999.

R. A. Brooks, *"A robust layered control system for a mobile robot"*, IEEE Journal of Robotics and Automation, p. 14-23, 1986.

Chaib-Draa B., Jarras I., Moulin B., *"Systèmes multi-agents : principes généraux et applications ,Principes et architectures des systèmes multi-agents*, J.-P. Briot et Y.Demazeau (Eds.) Hermès - 2001.

Beer M., d'Inverno M., Luck M., Jennings N., Preist C., Schroeder M., *"Negotiation in Multi-Agent Systems"*, Workshop of the UK Special Interest Group on Multi-Agent Systems (UKMAS'98) - 1998

L. Parker, *"Adaptative heterogeneous multi-robot teams"*, Neurocomputing, special issue of NEURAP'98: Neural networks and their applications, 28, p.75-92, 1999.

Drogoul A, *"De la simulation multi-agent à la résolution collective de problèmes. Une étude de l'émergence de structures d'organisation dans les systèmes multi-agents."*, Thèse de doctorat, Université Paris 6. - 1993

P. Lucidarme and A. Liégeois, *"Apprentissage de comportements réactifs pour des ensembles de robots mobiles"*,JJCR17, Paris, pp. 33-39, IEEE best paper award - 2003

P. Lucidarme, O. Simonin et A. Liégeois, *"Implementation and evaluation of a satisfaction/altruisme-based architecture for multi-robot systems"*, proc. ICRA'02, Washington D.C., pp. 1007-1012 - 2002

R. Alami, S. Fleury, M. Herrb, F. Ingrand and F. Robert *"Multi-robot cooperation in the Martha project"*, IEEE Robotics and AutomationMagazine, 5(1), p.36-47, 1998.

Pierre Tournassoud, *"Planification et contrôle en robotique Application aux robots mobiles et manipulateurs"*, Edition Hermes Traité des nouvelles Technologies série robotique – 1992

MITUL SAHA, *"Manipulator Planning and Obstacle Avoidance Using Genetic Algorithm"*, http://members.rediff.com/mitulsaha/ga.html

Dominique Duhaut, *"Distributed Algorithm For High Control Robotics Structures"*, International Conference On Artificial Intelligence – 1999 – Volume 1 p 45 à 50

Dominique Duhaut, *"Using multi agent approach solve inverse kinematic problem"*, proceeding 1993

Z. Guessoum, *"A Hybrid Agent Model: a Reactive and Cognitive Behavior"*, ISAD97, IEEE, Berlin, Germany, pp. 25-32 - April 1997.

A DISCRETE-EVENT SYSTEM APPROACH TO MULTI-AGENT DISTRIBUTED CONTROL OF CONTAINER TERMINALS

Guido Maione
DEESD, Technical University of Bari, Viale del Turismo, 8 - 74100, Taranto, Italy
gmaione@poliba.it

Keywords: Multi-Agent Systems, Discrete Event Systems, Container Terminals, Transport Systems.

Abstract: The area of modelling and controlling intermodal terminal systems is relatively new. The paradigms of Discrete Event Systems for modelling and of Multi-Agent Systems (MAS) for distributed control seem promising. Many research attempts developed modelling and simulation tools but no standard exists. This paper presents a Discrete Event System model of the agents introduced to describe how a distributed control of the terminal activities can be achieved. The interactions between four classes of agents are detailed. The approach is useful to develop a simulation platform to test MAS efficiency in terminal management and to measure the performance of static or adapted control strategies.

1 INTRODUCTION

Planning and scheduling resources in a maritime container terminal pose very complex modelling and control problems to the scientific community. Flexible and powerful modelling and simulation tools are necessary to describe an intermodal hub system made up of many different infrastructures and services. In the absence of standard tools, several research studies are based on discrete event simulation techniques (Vis and De Koster, 2003), on mathematical models or empirical studies (Crainic *et al.*, 1993, Gambardella *et al.*, 1998).

In the context of intelligent control of transport systems, this paper uses the Discrete EVent System (DEVS) specification technique (Zeigler *et al.,* 2000) to completely and unambiguously characterize a Multi-Agent System (MAS) for controlling a container terminal. Autonomous agents play as atomic dynamic DEVS. They exchange messages one with another to negotiate services in a common environment. The DEVS approach leads to heterarchical MAS where all information and control functions are distributed across agents, and allows rigorous theoretical analysis of structural properties of the MAS. Moreover, the DEVS formalism is an interesting alternative to other tools for MAS specification (Huhns and Stephens, 2001, Lin and Norrie, 2001). It is suitable to develop models both for simulation and for implementation of the software controllers.

As in MAS for manufacturing control (Heragu, 2002), agents can use decision algorithms based on a fictitious currency to buy services from other seller agents which use pricing strategies. Sellers and buyers reach an equilibrium between conflicting objectives, i.e. to maximize profit and to minimize costs, respectively. Recent analytical models of negotiation (Hsieh, 2004) underline the need of a systematical analysis and validation method for distributed networks of autonomous agents. Other researches focus on the experimental and detailed validation of MAS on distributed simulation platforms (Shattenberg and Uhrmacher, 2001, Logan and Theodoropoulos, 2001).

Here, a detailed DEVS model is developed for the interactions between the agents concurrently operating for the critical downloading process of containers from a ship. This paper is a contribution to define a complete DEVS model to develop a simulation platform for testing and comparing the proposed MAS with other distributed control architectures for container terminals. The simulation model could be used to design and test alternative system layouts and different control policies, both in standard and perturbed operating conditions.

The paper is organized as follows. Section 2 describes the main processes in a terminal. Then, the basic components of the MAS are presented, and their roles and relations are indicated. Section 3 models agents as atomic DEVS, and focuses on their interactions when containers are downloaded from

ships to the terminal yard. Section 4 gives some ideas about the simulation platform to test efficiency and robustness of the proposed MAS architecture. Section 5 overviews the benefits of the approach and enlightens open issues.

2 THE MAS FRAMEWORK FOR CONTAINER TERMINALS

To define the autonomous agents operating and interacting in a terminal environment, the main terminal processes have to be abstracted. They are commonly identified as import, export, and transshipment cycles.

2.1 Import, Export and Transshipment Cycles in a Container Terminal

Imported containers arrive on a vessel or feeder ship and depart by trains/trucks. Exported containers arrive by trains/trucks and depart on a vessel ship (transition from railway/road to sea mode). Transshipped containers arrive and depart by ship: cargo is moved from vessel (feeder) to feeder (vessel) ships for close (far) destinations.

Then, containers have to be: loaded/downloaded on/from ships; delivered/picked to/from trucks or trains; stacked and kept in blocks, in which the terminal yard is divided; transferred from ship/train/truck to blocks and backwards; consolidated, i.e. redistributed between blocks to allow fast retrieval. To this aim, several dedicated or shared resources are used: quay cranes, yard cranes, railway cranes; internal transport vehicles (trailers); quays and berths, lanes for internal transport, gates, railway tracks; skilled human operators.

A road import cycle follows three steps. Firstly, quay cranes download containers from a berthed ship to trailers, which transfer cargo to yard. Here, yard cranes pick-up containers from trailers and stack them in assigned positions. Secondly, containers stay in the blocks while waiting for their destination; they are eventually relocated by yard cranes and trailers in a more proper position. Thirdly, containers are loaded from blocks to trucks, and exit from the gate. Similarly, in a railway import cycle, yard cranes pick-up containers from blocks and load them on trailers moving from yard to the railway connection, where special cranes pick-up containers to put them on departing trains.

In a road/railway export cycle, the sequence goes in the opposite sense: from terminal gate or railway connection to blocks and then to vessel ships.

In a transshipment cycle, when a vessel ship arrives, containers are downloaded to trailers by quay cranes, transferred to blocks by trailers, picked-up and stacked by yard cranes. After a consolidation and/or a delay in their position, containers are picked-up and transferred to the quay area, where they are loaded on a feeder ship. The opposite occurs if a feeder arrives and a vessel departs.

Here, the focus is on the first step of the import and transshipment cycles. Similar models can be defined for the following steps and the export cycle.

2.2 Agents in a Container Terminal

The main agents controlling a terminal are:

- the *Container Agent* (CA): each CA is an autonomous entity controlling the flow of a single or a group of containers (stowed in a ship bay or stacked in yard blocks);
- the *Quay crane Agent* (QA): it is an autonomous controller for a (set of) quay crane(s) with the same performances or the same reachable ship bays;
- the *Trailer Agent* (TA): it is associated to a (set of) trailer(s) with the same performances, or with the same reachable quay or yard spaces;
- the *Yard crane Agent* (YA): it manages a (set of) yard crane(s) with the same performances, or associated to the same yard blocks;
- the *Railway crane Agent* (RA): it controls a (set of) railway crane(s), to receive containers from trailers and load them on trains at the railway connection, or to deliver containers from trains to trailers;
- the *Truck Agent* (KA): it follows the operations executed by a (set of) truck(s), entering or leaving the terminal by its gate.

In import processes, the CA identifies the most suitable quay crane to download containers from ship, then it selects the trailers to transport containers to their assigned yard blocks, and finally the yard cranes to pick-up and stack containers in their assigned block positions. In export processes, the CA selects the yard cranes to pick-up containers from blocks, trailers to transport them to the quay area, and quay cranes to load them into their assigned bay-row-tier location in the ship.

The decisions taken by a CA are based upon real-time updated information received from agents of the alternative available cranes and trailers.

The global control of the activities in the terminal emerges from the behaviour of concurrently operating agents. The dynamical interaction between agents has to be analysed to specify the desired

system behaviour. The precedent observations lead us to examine interactions between a CA and several QAs, TAs, and YAs for downloading, transferring, and stacking containers, or for picking, transferring, and loading containers. Interactions exist also between a CA and YAs, TAs, and RAs/KAs when railway/road cycles are considered.

The agents' decisions are only limited by constraints of terminal spaces and resources (e.g., the limited number of quay cranes that can serve a fixed ship bay; the limited number of yard cranes), and of schedules for downloading/loading processes (establishing the number of containers moved for each ship bay, the location in the ship hold or cover, sequence of handling moves, the preferred crane).

Agents' decisions are fulfilled by human operators devoted to the associated resources (e.g. crane operators, trailer/truck drivers). The network of interacting agents may appear and behave as a unique distributed supervisor for the terminal.

3 DEVS MODELLING OF AGENTS' DYNAMICS

Each agent in the previously identified classes is described as an atomic DEVS. Namely, all agents interact by transmitting outputs and receiving inputs, which are all considered as event messages. Events are instantaneous, then timed activities are defined by a start-event and a stop-event.

For each agent, internal events are triggered by internal mechanisms, external input events (i.e. inputs) are determined by exogenous entities (other agents), and external output events (i.e. outputs) are generated and directed to other entities.

External or internal events change the agent state. An agent stays in a state until either it receives an input or the time specified before an internal event elapses. In the first case, an external transition function determines the state next to the received input; in the second case, an internal transition function gives the state next to the internal event. An output function generates the reactions of the agent.

The sequential state, **s**, refers to the transition mechanism due to internal events, and is based on the current value of *status* (condition between consecutive events) and other information *i* peculiar to the considered agent: **s** = (*status*, *i*). The total state **q** = (**s**, *e*, **DL**), where *e* is the time elapsed since the last transition, and **DL** is the decision logic currently used by the agent to rank and choose the offers received by other agents.

For a CA, **s** may include information on: the current container position; the quay cranes available for negotiating handling operations; the trailers available for negotiating transport between quay and yard; the yard cranes available for negotiating handling operations from trailer to a block position or backwards; time scheduled in current state before the next internal event, if no input occurs.

For QAs, YAs, TAs, RAs, KAs, the sequential state may include the queued requests coming from CAs (for availability, for data about offered service, for confirmation of assigned service, etc.), and the time prospected before the next internal event.

To summarize, each agent can be represented as an atomic DEVS in the following way:

$$\boldsymbol{A} = < \mathbf{X}, \mathbf{Y}, \mathbf{S}, \delta_{int}, \delta_{ext}, \lambda, ta > \quad (1)$$

where **X** is the set of inputs, **Y** is the set of outputs, **S** is the set of sequential states, δ_{int}:**S**→**S** is the internal transition function, δ_{ext}:**Q**×**X**→**S** is the external transition function, $\mathbf{Q}=\{\mathbf{q}=(\mathbf{s},e,\mathbf{DL})|\mathbf{s}\in\mathbf{S},0\le e\le ta(\mathbf{s})\}$, λ:**S**→**Y** is the output function, ta:**S**→$\Re_0^+$ is the time advance function, with $\Re_0^+$ set of positive real numbers with 0 included.

The negotiation between agents is typically organized in the following steps. *Announcement*: an agent starts a bid and requires availability to other agents for a service. *Offer*: the agent requests data only to the available agents. Data regard the offered service the queried agents can guarantee. *Reward*: the agent selects the best offer between the collected replies and sends a rewarding message. *Confirmation*: the agent waits for a message from the rewarded agent, then it acquires the service. If confirmation does not arrive, then the agent selects another offer in the rank or restarts the bid.

In this paper, the focus is on the interactions between a CA with QAs, TAs and YAs for downloading containers from ship to a yard block. The study can be easily extended to other negotiations for loading or other processes. Besides, the *status* transitions are examined since the *status* is the main component of the sequential state.

3.1 Dynamics of Interactions between Agents in a Downloading Process

To download containers from a ship bay, transport and stack them into a yard block, each CA interacts with QAs to choose the quay crane for downloading, with TAs to select the trailers for moving the containers from the quay to the yard area, and finally with YAs to determine the yard cranes for stacking.

We assume that the CA firstly communicates exclusively with QAs, then with TAs only, finally with YAs. The negotiation is based on data about the offered services: a QA gives the estimated time to wait before the related crane can start downloading containers, and the estimated time to execute the task; a TA the estimated times to wait a trailer and for the transport task; a YA the estimated time for the yard crane to be ready close to the block, and the estimated time for the stacking task.

3.1.1 Interactions between a CA and QAs

For $t<t_{C0}$ let the CA, say *C*, associated with a generic container be in a quiescent *status* (QUIESC) and let it begin its activity at t_{C0} (input X_{C0}). Then *C* spends the interval $[t_{C0},t_{C1}]$ to send outputs Y_{C01}, Y_{C02},…, Y_{C0q} at instants $t_{01}>t_{C0}$, t_{02},…, $t_{0q}=t_{C1}$. These messages request the availability to all the *q* alternative QAs of cranes that can serve the container. The sequence of requests (REQQAV) is not interrupted by any internal or external event. *C* makes transition at t_{C1} (internal event I_{C1}).

In $[t_{C1},t_{C2}]$ *C* waits for answers (WAIQAV) from QAs. Namely, the request *C* transmits to each QA may queue up with similar ones sent by other CAs. Next transition occurs at t_{C2} when either *C* receives all the answers from the queried QAs (X_{C1}), or a specified time-out of WAIQAV expires before *C* receives all the answers. In case it receives no reply within the time-out (I_{C2}), *C* returns to REQQAV and repeats the request procedure. In case of time-out expiration and some replies received (I_{C3}), *C* considers only the received answers to proceed. The repeated lack of valid replies may occur for system congestion, crane failures, communication faults, or other unpredictable circumstances. To avoid indefinite circular waits and improve system fault-tolerance, we use time-outs and let *C* repeat the cycle REQQAV-WAIQAV only a finite number of times, after which *C* is replaced by another agent.

If all or some replies arrive before the time-out expiration, *C* starts requesting service to the $g \leq q$ available QAs at t_{C2}. In $[t_{C2},t_{C3}]$ *C* requests information with Y_{C11}, Y_{C12}, …, Y_{C1g} at instants $t_{11}>t_{C2}$, t_{12},…, $t_{1g}=t_{C3}$. The uninterrupted sequence of requests in *status* REQQSE terminates at t_{C3} and *C* makes transition (I_{C4}).

Then, *C* spends $[t_{C3},t_{C4}]$ waiting for offers from the available QAs (WAIQOF), as the request *C* transmits to each QA may queue up with those sent by other CAs. Next transition occurs at t_{C4} when either all the answers from the QAs arrive (X_{C2}) or a time-out of WAIQOF expires. In case no reply comes within the time-out (I_{C5}), *C* returns to REQQSE and restarts. If time-out expires and some replies are received (I_{C6}), *C* considers only the received offers. Again, cycling between REQQSE and WAIQOF is only for a finite number of times.

Once received the offers from QAs, *C* uses $[t_{C4},t_{C5}]$ to take a decision for selecting the quay crane (TAKQDE). At t_{C5} the decision algorithm ends (I_{C7}): all the offers are ranked and a QA is selected.

Subsequently, *C* reserves the chosen crane with message Y_{C2} to the corresponding QA. *C* takes $[t_{C5},t_{C6}]$ for communicating the choice to the winner QA (COMCHQ). At t_{C6} the communication ends (I_{C8}). Now, the QA sends X_{C5}: a rejection, if there is a conflict with another CA, or a confirmation. Hence, *C* uses $[t_{C6},t_{C7}]$ to wait for a confirmation from the selected QA (WAIQCO). The confirmation is necessary because CAs different from *C* act during the decision interval, and the selected crane can be no longer available. If *C* receives a rejection (X_{C3}), or no reply within a time-out (I_{C9}), it returns to COMCHQ, sends a new request of confirmation to the second QA in its rank. If *C* has no alternative destinations and the rejection (X_{C4}) or the time-out (I_{C10}) occurs, it returns to REQQAV and restarts.

At t_{C7}, after receiving a confirmation (X_{C5}) from the selected QA, *C* makes a transition to DWNLDG: interval $[t_{C7},t_{C8}]$ is for issuing the command Y_{C3} to the quay crane downloading the container.

Figure 1 depicts the complex dynamics previously described. Circles represent *status*-values and encapsulate outputs, arrows represent internal or input events.

3.1.2 Other Interactions

When, at time t_{C8}, the command is complete (I_{C11}), *C* starts the negotiation with TAs for a trailer.

C follows the same procedure as with QAs (requests and waits for availability and then for offers, decision for the best offering TA, waits for a confirmation/rejection). Then, the graph in figure 1 continues with a second part (not shown for lack of space), with the same structure as the first part. This guarantees modularity, which is important for simulation and control. After a confirmation, *C* issues a transport command, and the container is loaded on the vehicle associated to the selected TA.

Then, *C* starts the negotiation with YAs to stack the container in an assigned block position. Again, due to the modularity of the approach, the sequence of *status*-values follows the same protocol, and, finally, if a confirmation is received, *C* issues a stacking command and gets back to QUIESC.

C stops interacting and remains quiescent until the beginning of the next negotiation (if any) for downloading, consolidating or loading another container. The associated container is downloaded, transported and stacked in a block where it waits for the next destination (a block or a ship). If faults

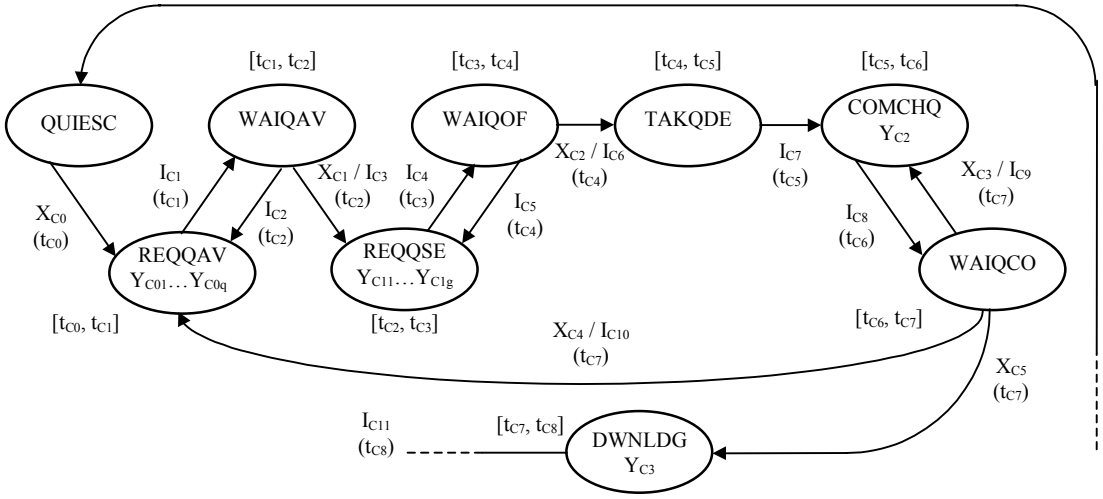


Figure 1: Dynamics of the interaction between a CA when negotiating with QAs.

occur to the selected cranes or trailer, *C* remains in QUIESC. Terminal operators restore the normal operating conditions and the container can be handled by the selected resources.

3.2 Specification of the DEVS Model of a Container Agent

We may identify the components of the DEVS model of a CA, on the basis of the negotiation mechanism previously described. Table 1 reports the admissible *status*-values. Detailed description of inputs, outputs and internal events is not reported for lack of space, but it can be easily derived from the interactions between a CA and QAs, TAs, and YAs.

The sequential state **s** collects the *status*-value and other information *i* about the CA, which is:

$$i = (p, AQ, AT, AY, ta(\mathbf{s})) \qquad (2)$$

including the current position *p* of the container (on ship, picked by quay crane, on trailer, picked by yard crane, in the yard block); the set *AQ* of quay cranes available for the currently negotiated downloading operations, or the set *AT* of trailers available for the currently negotiated transport, or the set *AY* of yard cranes available for currently negotiated stacking tasks; the time *ta*(**s**) scheduled in current state before the next internal event.

The time advance function gives the residual time *ta*(**s**) in state **s** before the next scheduled internal invent. E.g., at the time t^* of entering a waiting status, the time-out fixes T_w, such that $ta(\mathbf{s}) = t^*+T_w-t$, where t is the current time.

Table 1: *Status*-values for a Container Agent.

Status	Activity Description
QUIESC	Agent quiescent
REQQAV	Request availability to all QAs
WAIQAV	Wait for availability from QAs
REQQSE	Request service to available QAs
WAIQOF	Wait for offers from available QAs
TAKQDE	Take decision for the best QA
COMCHQ	Communicate choice to selected QA
WAIQCO	Wait conf./reject. from selected QA
DWNLDG	Command selected QA to download
REQTAV	Request availability to all TAs
WAITAV	Wait for availability from TAs
REQTSE	Request service to available TAs
WAITOF	Wait for offers from available TAs
TAKTDE	Take decision for the best TA
COMCHT	Communicate choice to selected TA
WAITCO	Wait conf./reject. from selected TA
TRANSP	Command selected TA to transport
REQYAV	Request availability to all YAs
WAIYAV	Wait for availability from YAs
REQYSE	Request service to available YAs
WAIYOF	Wait for offers from available YAs
TAKYDE	Take decision for the best YA
COMCHY	Communicate choice to selected YA
WAIYCO	Wait conf./reject. from selected YA
STCKNG	Command selected YA to stack

DEVS models can be also specified for the other agents, and *status*-transition diagrams can be defined for the interactions occurring when agents negotiate tasks in loading or consolidation processes.

4 IDEAS FOR SIMULATING MAS

The DEVS atomic models can be integrated in a network, which can be used as a platform for simulating the MAS controlling a terminal, e.g. the Taranto Container Terminal (TCT).

In this context, it is possible to simulate not only the dynamics of terminal activities, the flow of containers, and the utilization of terminal resources (cranes, trailers, human operators, etc.), but also the efficiency of the MAS and its agents (flow of event messages, *status* transitions, waiting loops, etc.).

Then, two types of performance indices can be defined. Namely, it is possible to measure conventional indices: the total number of (imported, exported, transshipped) containers; the average throughput, during downloading (from ship to yard) or loading (from yard to ship) processes; the average lateness of containers in the terminal. Moreover, it is possible to measure the behaviour of the MAS and the efficiency of the agents' decision policies by means of: the average number of requests for each negotiation; the number of loops of *status*-values before a final decision is taken by a CA, expressed in percentile terms with respect to the total number of operations executed by every CA.

The performance measures can be evaluated both in steady-state operating conditions and in perturbed conditions. Perturbations may arise from: hardware faults or malfunctions; abrupt increase/decrease of maritime traffic volumes; sudden increase/reduction of yard space; traffic congestion of trailers; congestion, delays, message losses, and faults in the communication between agents.

Then, it is important to measure robustness of agents' decision laws, to see how they dynamically react to disturbances and parameter variations, and eventually to adapt them. The adaptation aims to make the autonomous agents learn the most appropriate decision laws in all terminal conditions.

To conclude, the simulation platform will allow to compare control architectures defined by:

- a static MAS in which CAs use heuristic decision parameters (estimated time of the requested task, distance of cranes or trailers);
- a dynamic MAS in which CAs take decisions by fuzzy weighted combinations of heuristic decision criteria; the weights can be adapted by an evolutionary genetic algorithm.

5 CONCLUSIONS

This paper proposes a MAS architecture for controlling operations in intermodal container terminal systems. The autonomous agents are represented as atomic DEVS components. The interactions between agents are modelled according to the DEVS formalism to represent negotiations for tasks when downloading containers from ship to yard stacking area. The developed model can be easily extended to describe other processes (loading containers from yard area to ships, redistributing containers in the yard area).

The DEVS model of the MAS can be used in a detailed simulation environment of the TCT, which allows to measure standard terminal performance indices and the efficiency of the MAS. Moreover, open issues are testing and comparing static MAS and dynamically adapted MAS, if evolutionary adaptation mechanisms are used.

REFERENCES

Crainic, G., Gendreau, M., Dejax, P., 1993. Dynamic and stochastic models for the allocation of empty containers. *Oper. Research*, Vol. 41, pp. 102-126.

Gambardella, L.M., Rizzoli, A.E., Zaffalon, M., 1998. Simulation and planning of an intermodal container terminal. *Simulation,* Vol. 71, No. 2, pp. 107-116.

Heragu, S.S., Graves, R.J., Kim, B.-I., Onge, A.St., 2002. Intelligent Agent Based Framework for Manufacturing Systems Control. *IEEE Trans. Sys., Man, and Cyber. - Part A*, Vol. 32, No. 5.

Hsieh, F.-S., 2004. Model and control holonic manufacturing systems based on fusion of contract nets and Petri nets. *Automatica*, 40, pp. 51-57.

Huhns, M.N., Stephens, L.M., 2001. Automating supply chains. *IEEE Int. Comput.*, Vol. 5, No. 4, pp. 90-93.

Lin, F., Norrie, D.H., 2001. Schema-based conversation modeling for agent-oriented manufacturing systems. *Computers in Industry*, Vol. 46, pp. 259-274.

Logan, B., Theodoropoulos, G., 2001. The distributed Simulation of Multiagent systems. *Proceedings of the IEEE*, Vol. 89, No.2, pp. 174-185.

Shattenberg, B., Uhrmacher, A.M., 2001. Planning Agents in James. *Proc. of the IEEE*, Vol. 89, No. 2, pp. 158-173.

Vis, I.F.A., De Koster, R., 2003. Transshipment of containers at a container terminal: An overview. *Europ. Jour. of Oper. Research*, Vol. 147, pp. 1-16.

Zeigler, B.P., Praehofer, H., Kim, T.G., 2000. *Theory of Modelling and Simulation*, Academic Press. New York, 2nd ed..

TRACKING A WHEELCHAIR WITH A MOBILE PLATFORM

B.Allart, B. Marhic, L. Delahoche, A. Clérentin
Groupe RTeAM of UPJV – IUT Informatique Avenue des facultés – Le Bailly - 80025 Amiens Cedex 1, France
Benjamin.allart@cegetel.net, bruno.marhic@u-picardie.fr, Laurent.delahoche@u-picardie.fr
Arnaud.clementin@u-picardie.fr

O. Rémy-Néris
CHU of Brest, 5 avenue Foch 29200 Brest
olivier.remy-neris@wanadoo.fr

Keywords: Target tracking, Kalman filter, camshift, mobile plateform.

Abstract: This article deals with a target tracking application for the disabled. The objective of this work is to track a wheelchair with a mobile platform and an embedded grasping arm (MANUS). We propose an approach based on an association of two Kalman filtering levels. The first level permits an estimation of the wheelchair configuration. The second is used to compute the mobile platform configuration in connection with its environment. The association of the two filtering process allows a robust tracking between two objects in movement.

1 INTRODUCTION

In this article we propose an original approach to solve the problem of configuration estimation of a target observed by a robot in movement. We propose a probabilistic approach based on Kalman Filtering. The problem of tracking is classical in the world of robotics. It's generally linked to the data association stage. The data association problem is that of associating the many measurements made by a sensor with the underlying states or trajectories that are being observed. It includes issues of validating data, associating the correct measurement to the correct states or trajectories, and initializing, confirming or deleting trajectories or states. This way, the Probabilistic Data Association Filter (PDAF) (Y. Bar Shalom et T.E. Fortmann, 1988) for single targets and the Joint Probability Data Association Filter (JPDAF) (Y. Bar Shalom et T.E. Fortmann, 1988 and Bar-Shalom Y, Xiao-Rong Li,1995) for multiple targets are two inescapable approaches. They are both Bayesian algorithms that compute the probability of correct association between an observation and a trajectory. We can combine the Sequential Monte Carlo method to decline the JPDAF method.

A second classical paradigm of data association is the Multiple hypothesis tracking (MHT) which permits to represent multimodal distributions with Kalman filters (Y. Bar Shalom et T.E. Fortmann, 1988). It has been used with great effectiveness in radar tracking systems, for example. This method maintains a bank of Kalman filters, where each filter corresponds to a specific hypothesis about the target set. In the usual approach, each hypothesis corresponds to a postulated association between the target and a measured feature.

For our application, we have made the choice to use two Kalman filters to solve the problem of tracking between two objects in movement.

In a first part we present the used perception system, which permits to track the wheelchair that is to say a stereo omnidirectional sensor.

In a second part, we address the problem of wheelchair recognition using vision sensors.

In the third part, we deal with the multi-level Kalman filtering tracking.

We conclude with an explanation of experimental results.

2 THE MOBILE PLATFORM

2.1 Context Overview

This work deals with technical assistance for persons of reduced mobility. The mobile platform is built with a wheelchair frame. The reader interested by

this robotic assistance can find details in (B. Marhic, L. Delahoche ,F. de Chaumont, and O. Remy-Néris, 2006).

The SPI group of the IUT of Amiens has applied its skills in the domains of mobile robotics and detection of the surrounding environment. It is involved in the integration of a system of detection via a motorised platform that can be mounted by a grasping arm MANUS (R).

2.2 Sensors Involved in this Paper

2.2.1 Dead Reckoning and its Uncertainty

We are going to establish the discontinuous equations of the platform position considering small displacements (Figure 1)

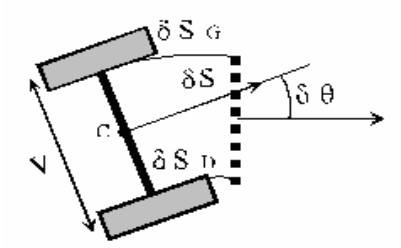


Figure 1 : Displacement of the robot during a period.

We thus obtain: $q(n+1) = q(n) + \Delta q(n)$ where q is the position of the mobile platform.

In which:

$$\Delta q(n) = \begin{bmatrix} \delta S(n) \cdot \frac{\sin(\delta\theta(n)/2)}{\delta\theta(n)/2} \cdot \cos(\theta(n) + \delta\theta(n)/2) \\ \delta S(n) \cdot \frac{\sin(\delta\theta(n)/2)}{\delta\theta(n)/2} \cdot \sin(\theta(n) + \delta\theta(n)/2) \\ \delta\theta(n) \end{bmatrix} \quad (2)$$

The vector from the equation (2) which is exact, can be simplified if we consider that the sampling period is small enough (sin(δθ (n)/2)/ (δθ (n)/2))≈1.

Thus (order 0),

$$\Delta q(n) = \begin{bmatrix} \delta S \cdot \cos(\theta(n)) \\ \delta S \cdot \sin(\theta(n)) \\ \delta\theta \end{bmatrix} \quad (3)$$

The sampling period being very small, it is possible to assimilate the elementary displacement to a segment. Therefore we will use the development to the order 0 (3), into the following calculations:

$$\begin{cases} X_{odo}(n+1) = X_{odo}(n) + \delta S(n) \cdot \cos(\theta(n)) \\ Y_{odo}(n+1) = Y_{odo}(n) + \delta S(n) \cdot \sin(\theta(n)) \\ \theta(n+1) = \theta(n) + \delta\theta(n) \end{cases} \quad (4)$$

With the matrix form, we obtain (F is a non-linear function in *q):*

$$\begin{aligned} & q(n+1) = F\left(q(n), \Delta(n)\right) \\ & \Delta(n) = [\delta S(n), \delta\theta(n)]^t \\ & \Delta(n)_{mes} = \Delta(n) + Bq = \hat{\Delta}(n) \end{aligned} \quad (5)$$

$Bq \sim N(0, V^{bq})$ (Gaussian noises, centred)

We will apply the Taylor development of F (equation 5) around $\left(\hat{q}(n-1), \hat{\Delta}(n-1)\right)$, in order to render the equations linear.

2.2.2 Stereoscopic Omnidirectional Vision System

On the figure 2, we can see the configuration of the two omnidirectional vision sensors.

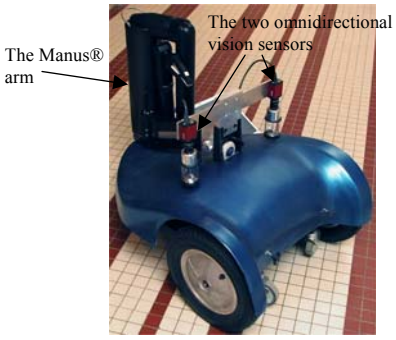


Figure 2: The mobile platform.

Main vision applications in mobile robotics use the classical pinhole camera model. Thus according to the lens used, the field of view is limited. Nevertheless, it is possible to enlarge the field of view by using cameras mounted in several directions (H. Ishiguro, S. Tsuji, 1993), but the information flow is very important and time consuming. Other applications (E. Colle, Y. Rybarczyk, P. Hoppenot, 2002) use only one camera, with a rotation motion, in order to sweep a large space. The disadvantage of such a system is that the camera's movement takes time; and what's more, a mechanical looseness can appear in the course of time. To get wide-angle pictures, another possibility exists: omnidirectional vision. These kinds of sensors allow acquiring scenes with 360° field of view (El. M. Mouaddib, B. Marhic, 2000). There are two major classes of omnidirectional vision systems. First of all, systems

made of a mirror and a camera, are called "catadioptric systems" (C. Cauchois, E. Brassart, L. Delahoche, T. Delhommelle, 2000)(H. Ishiguro, S. Tsuji, 1996). The second one is composed of a classical camera with a fish-eye lens; such mountings are called "dioptric systems" (Z. L. Cao, S. J. Oh, Ernest L. Hall, 1986). We focus on the first class.

There are many advantages to using an omnidirectional vision sensor. Firstly, in one acquisition, we obtain a full view of the environment without using a sophisticated mechanical system. Secondly, even if the interpretation of omnidirectional pictures is difficult for novices, we can easily compute a "classical perspective view" of the scene. Finally, providing a picture in a chosen direction is instantaneous.

The omnidirectional vision system we use is made of a digital color video camera and a hyperbolic mirror. Figure 3 shows an omnidirectional view of an environment with a wheelchair in the field of view.

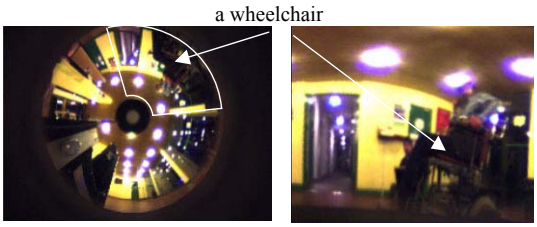


Figure 3: (left) an omnidirectional view of a scene with a wheelchair in the field of view. (right) "un-warped" picture of the white area from the omnidirectional view.

3 TRACKING RECOGNITION

3.1 Initialisation (Target-wheelchair)

We wished to achieve the greatest possible degree of flexibility regarding the use of the robotised assistance. We therefore did not want to restrict our method to the use of one wheelchair in particular.

Our construction of the model accommodates not only the wheelchair, but also the patient. This is why we turned our work towards an intrinsic polymorph (self re-configuring), directly calculated from a stereoscopic colour video signal. The figure below (Figure 4) shows omnidirectional images: they illustrate the extraction of the background and the extraction of the wheelchair.

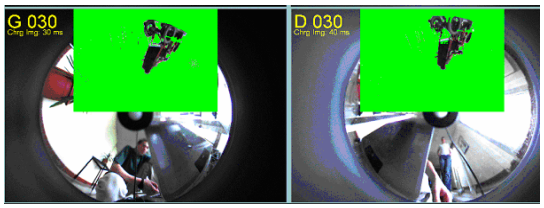


Figure 4: Target Initialisation.

Once the model is computed, a histogram representation is calculated.

3.2 The OmniCAMShift Recognition and Triangulation

As the wheelchair is not equipped with any particular marker, we have to track it as it is. This way, we use the CAMShift algorithm, which performs a tracking, by using an image of the object to track. The Continuously Adaptive Mean Shift (CAMShift) algorithm (C. Cauchois, E. Brassart, L. Delahoche, T. Delhommelle, 2000), is based on the mean shift algorithm (B. Marhic, L. Delahoche ,F. de Chaumont, and O. Remy-Néris, 2006), a robust non-parametric iterative technique for finding the mode of probability distributions including rescaling.

We have named "Omnicamshift" the calculation of a CAMShift directly in an omnidirectional image. We have also applied some specificity linked to the sensor used (fast gyration, …). The next figure (Figure 5) shows an example of the OmniCAMShift application:

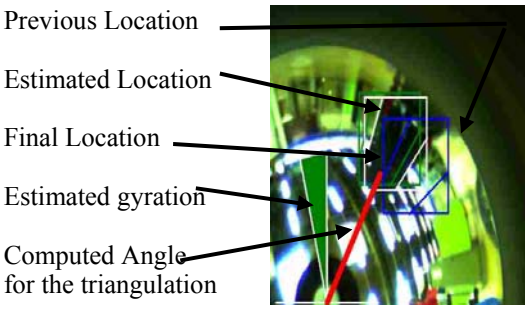
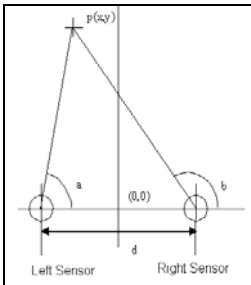


Figure 5: Wheelchair recognition using OmniCAMShift.

Once the wheelchair is identified in both omnidirectional images, computing the relative position of the wheelchair by triangulation is a minor task (Figure 6) :



$$x = \frac{d \times \tan(b)}{\tan(b) - \tan(a)} + \frac{d}{2}$$

$$y = \frac{d \times \tan(b) \times \tan(a)}{\tan(b) - \tan(a)}$$

Figure 6: triangulation.

4 TRAKING WITH KALMAN FILTER

4.1 The Kalman Filter

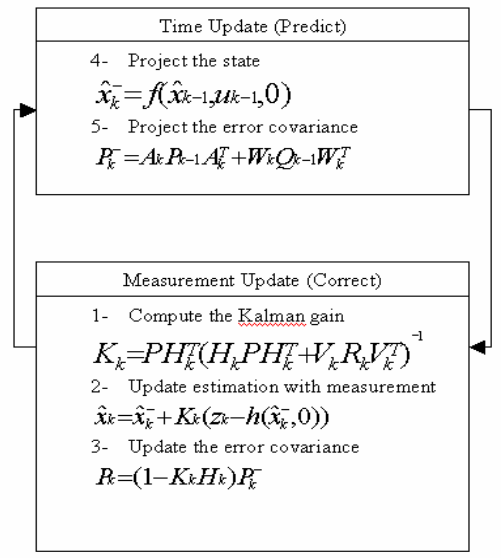


Figure 7: Kalman filter.

We would like to recall at this point that the process that manages the movement of the RMA is discontinuous and non linear in θ. The measurements Y(n) are linked to q(n) by an equation of observation (or equation of measurement).

$$\begin{aligned} q(n+1) &= F\big(q(n), \Delta(n)\big) + v \\ \Delta(n) &= \big[\delta S(n), \delta\theta(n)\big]^t \\ Y(n) &= H \cdot q(n) + w \end{aligned} \qquad (6)$$

with :

⇨ v is a centred Gaussian white noise, of the variance matrix Cpro.

⇨ w is a Gaussian white noise which perturbs the measurement.

⇨ Δ(n) is the vector of command.

⇨ q(n) is the trajectory of the state vector, representing the localisation of the mobile platform.

⇨ H is the observation matrix.

4.2 Filtering with Non-inner Observation

In this application, we have two objects in movement, a wheelchair and a mobile platform. However, we only have proprioceptive movement information on the mobile platform. Using dead-reckoning we compute the position of our mobile platform and using the omnidirectional vision system (exteroceptive), we calculate the relative position of the wheelchair compared to our mobile platform.

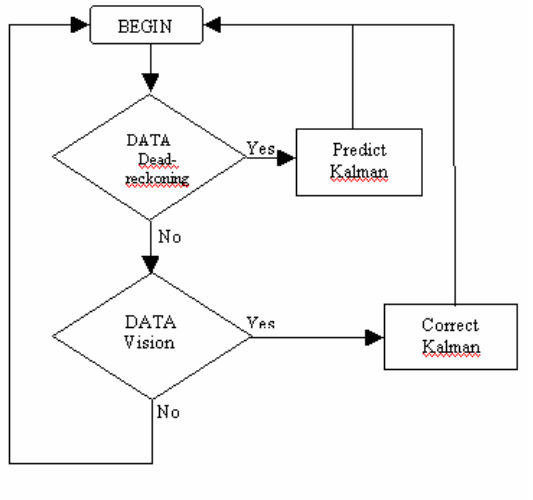


Figure 8: Filter's algorithm.

From the implementation of our model, we decided to use for the prediction of the Kalman filter the data give by the odometric sensor and for the step of update the data give by triangulation of the two omni-directional sensor.

We decide to make the triangulation operation out of your filter and injected directly the result in the update step of kalman filter. That makes it possible to bring us back to a linear system.

The vision module permits to obtain $Xf = \begin{bmatrix} x_f & y_f \end{bmatrix}^t$, where :

$$X_f = X_{odo-1} + \Delta D_n * \cos(\theta_{n-1} + \frac{\Delta\theta_n}{2}) + X_{tri} \qquad (7)$$

$$Y_f = Y_{odo-1} + \Delta D_n * \sin(\theta_{n-1} + \frac{\Delta\theta_n}{2}) + Y_{tri} \qquad (8)$$

with Xtri and Ytri the position between the mobile platform and the wheelchair and, X_{odo} and Y_{odo} the result of odometric equations .

In order to have a homogeneous filtering, the vision uncertainty of the localisation is considered to be a Gaussian white noise.

Thus, we obtain :

$$\begin{aligned} Y &= Xf + w \\ H &= I \end{aligned}$$

"*I*" being the identity matrix.

The result obtained was satisfactory for straight lines (figure 9) but insufficient during the phase where the mobile platform turned due to errors of odometry.

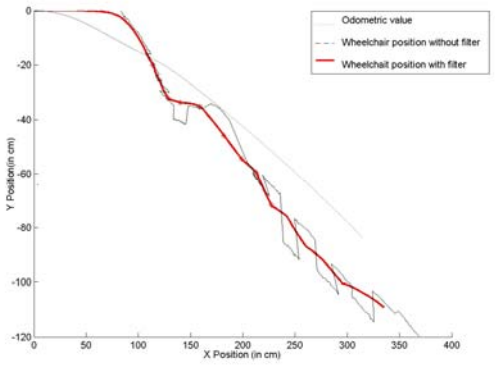


Figure 9: Filter with a good precision.

4.3 Extended Filtering

To remove the problem of the imperfection of the dead-reckoning, we will use a method which requires a knowledge of the landmarks. We will be able to determine with precision, the position of our mobile plateform and to thus replace it to avoid the errors of the dead-reckoning.

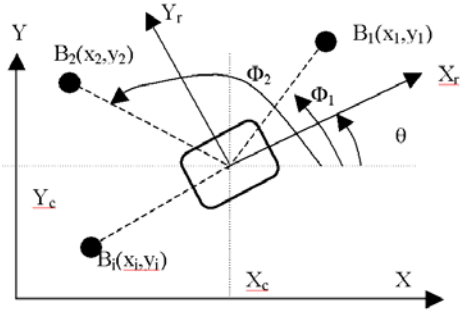


Figure 10: Relation between beacon and mobile platform.

For us, these landmarks are walls, doors, objects, angles which one will be able to detect in an omnidirectional image. Therefore, it is necessary for us to know the map of the environment to be able to mark it out.

Moreover, this method is based on an extended Kalman filter that can be integrated into our preceding process.

For this process, the equation of observation of the extended Kalman is as follows.

The vector of observation is written:

$$z_k^* = \begin{bmatrix} 1_k^\tau \\ 2_k^\tau \\ . \\ . \\ n_k^\tau \end{bmatrix} + v_k = h(X_k, k) + v_k \qquad (9)$$

where i_k^τ is the layer of i^{ème} beacon B_i of co-ordinates (x_i, y_i) i the world landmark in the moment k. And v_k is a measurement noise, presumably white and Gaussian.

The exact position of the beacon B_i is expressed according to the state vector X_k of the system as follows:

$$i_k^\tau = \arctan\left(\frac{y_k - y_i}{x_k - x_i}\right) \qquad (10)$$

The matrix of the Jacobien of the vector function H is, in the case of measurements of absolute angle:

$$H_k = \begin{bmatrix} -(y_k - y_1)/_1 d_k^2 & -(x_k - x_1)/_1 d_k^2 & 0 \\ . & . & . \\ -(y_k - yn)/_n d_k^2 & -(x_k - x_n)/_n d_k^2 & 0 \end{bmatrix}_{X_k = \hat{X}_{k/k-1}} \qquad (11)$$

where *d* is the distance between the landmark and the mobile platform

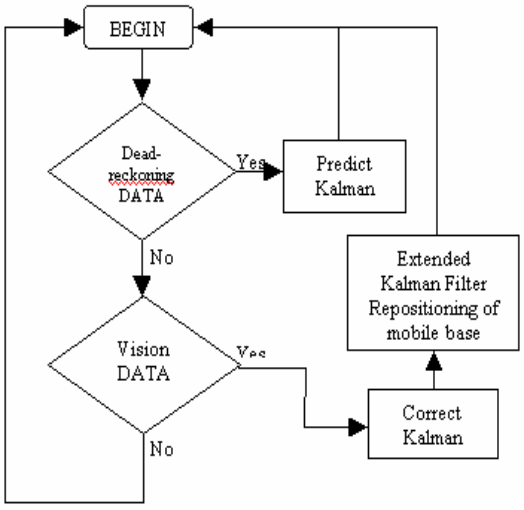


Figure 11: Process of filtering.

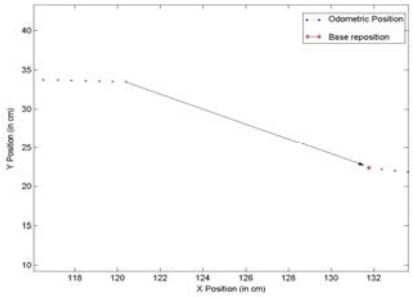


Figure 12: Repositioning of the mobile platform.

Table 1: Data after simulation of the system.

Reel base position		Estimate base position with kalman filter	
X (in cm)	Y (in cm)	X (in cm)	Y (in cm)
-166	-950	-168	-950
-240	-950	-240	-950
-327	-949	-327	-949
-365	-949	-365	-949
-400	-949	-419	-949
-463	-974	-463	-974

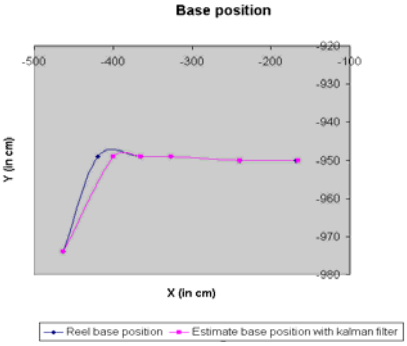


Figure 13: System's result.

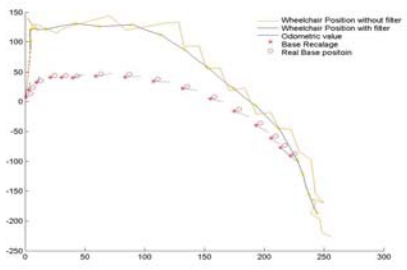


Figure 14: Other Result of the application.

With this method, we clearly see a marked improvement of the localisation, especially when the mobile platform turns. (Figure 12) which enabled us to improve the precision of our system (Table 1, Figure 13 where we can see the result of the system in a right line and Figure 14 which is the system's result in a turning).

5 CONCLUSION

In this article, we studied a target tracking application for the physically disabled. The aim is to track a wheelchair with a mobile platform mounted with a grasping arm (MANUS). We propose an approach based on an association of two Kalman filtering levels. The first level permits to estimate the wheelchair configuration. The second is used to compute the mobile platform configuration in connection with its environment. We have shown that the second level increases the robustness of the configuration estimation of the wheelchair in the platform frame. The use of the identity matrix in the first stage of the Kalman filtering permits to solve the problem of the non-linearity of the system.

This paradigm can be a contribution to finding a solution for tracking several objects in movement. The robustness of the filtering process is very important. Future works will study the integration of a supplementary layer based on a particle filter.

REFERENCES

Z. L. Cao, S. J. Oh, Ernest L. Hall. Omnidirectional dynamic vision positioning for a mobile robot" *Journal of Robotic System*, 3(1), 1986, pp5-17.

C. Cauchois, E. Brassart, L. Delahoche, T. Delhommelle. "Reconstruction with the calibrated SYCLOP sensor" *in Proc, Int. Conf. on Intelligent Robots and Systems*, Kagawa University, Takamatsu, Japan , pp. 1493-1498, October- November 2000.

E. Colle, Y. Rybarczyk, P. Hoppenot. "ARPH: An assistant robot for disabled person" *in Proc. IEEE International Conference on Systems, Man and Cybernetics*, Hammamet, Tunisia, October 6-9, 2002.

H. Ishiguro, S. Tsuji "Applying Panoramic Sensing to Autonomous Map Making a Mobile Robot" *in Proc, Int. Conf. on Advanced Robotics*, pp127-132, November 1993.

H. Ishiguro, S. Tsuji "Image-based memory of environment" *in Proc, Int. Conf. on Intelligent Robots and Systems*, pp634-639, Osaka, Japan, November 1996.

El. M. Mouaddib, B. Marhic, "Geometrical Matching for Mobile Robot Localisation". *IEEE Trans. Robotics and Automation*, vol. 16, n°5, pp 542-552, October 2000.

B. Marhic, L. Delahoche ,F. de Chaumont, and O. Remy-Néris, "Robotised Assistance for Persons of Reduced Mobility: résumé of a project", ICOST'2006, Ireland.

Y. BAR SHALOM et T.E. FORTMANN, "Tracking and data association", Academic Press, 1988.

Bar-Shalom Y, Xiao-Rong Li, Multitarget-Multisensor Tracking: Principles and techniques, 1995.

SETPOINT ASSIGNMENT RULES BASED ON TRANSFER TIME DELAYS FOR WATER-ASSET MANAGEMENT OF NETWORKED OPEN-CHANNEL SYSTEMS

Eric Duviella
Ecole des Mines de Douai, Dpt. Informatique et Automatique, 941 rue Charles Bourseul, 59508 Douai, France
duviella@ensm-douai.fr

Pascale Chiron and Philippe Charbonnaud
Laboratoire Génie de Production, Ecole Nationale d'Ingénieurs de Tarbes, 47 av. d'Azereix, 65016 Tarbes, France
{*pascale.chiron, philippe.charbonnaud*}*@enit.fr*

Keywords: Supervision, hybrid control accommodation, resource allocation, setpoint assignment, networked systems, water management.

Abstract: The paper presents a new strategy based on a supervision and hybrid control accommodation to improve the water-asset management of networked open-channel systems. This strategy requires a modelling method of the network based on a weighted digraph of instrumented points, and the definition of resource allocation and setpoint assignment rules. Two setpoint assignment rules are designed and evaluated in the case of an open-channel system composed of one difluent and one confluent showing their effectiveness.

1 INTRODUCTION

A hydrographic network is a geographically distributed system composed of dams and interconnected rivers and channels. Weather conditions and human activities have a great influence on the flow discharges. An interesting problem to address deals with the allocation of water quantities in excess toward the catchment area and of water quantities in lack amongst the users. The complex hydrographic network representation, as well as the determination of the discharge allocation on the network, constitute an essential step for the design of reactive control strategies. In (Naidu et al., 1997) a hydrographic network representation by oriented graphs is proposed by considering only the difluences. This representation is modified and extended to the cases of the confluences in (Islam et al., 2005). Cembrano *et al* (Cembrano et al., 2000) proposed a modelling approach for the drinking water distribution networks, and sewerage networks. Object-oriented modelling techniques (Chan et al., 1999) and a XML approach (Lisounkin et al., 2004) allow the representation of the control and measurement instrumentation equipping the hydrographic networks and the drinking water distribution networks. Optimization techniques were proposed in the literature for the water-asset management. The approach proposed in (Faye et al., 1998) allows the adjustment of the criteria and the constraints of an optimization problem starting from the supervision of the network variables. However, the complexity of the hydrographic networks and the number of instrumented points to be taken into account in the optimization problem require the use of decomposition and coordination techniques of the studied systems as proposed in (Mansour et al., 1998). These techniques are used for the optimal water management of irrigation networks. Finally, in (Duviella et al., 2007), a supervision and hybrid control accommodation strategy is proposed for the water asset management of the Neste canal in the southwestern region in France. This strategy can be adapted for the case of gridded hydrographic networks.

In this paper, the allocation and setpoint assignment rules are proposed for the water asset management of complex hydraulic systems *i.e.* with confluences and difluences. Networked hydraulic systems modelling is presented in section 2. In section 3, identification steps of transfer time delay are presented. The supervision and resource allocation rules are proposed in section 4. Section 5 deals with the design of a water asset management strategy where two setpoint assignment rules are compared. Finally, their evaluation by simulation within the framework of a hydrographic system is carried out.

2 NETWORKED HYDRAULIC SYSTEM MODELLING

Hydrographic networks are composed of a finite number of *Simple* Hydraulic Systems (HYS), *i.e.* composed of one stream. A HYS *source* is defined as a HYS which is not supplied by others HYS. A representation is proposed to locate the instrumentation, *i.e.* the sensors and the actuators, and to be able to determine the way to distribute a water quantity measured in a place of the hydrographic network, onto the whole HYS downstream. HYS are indexed by an index b, and all these indices forms the set $\mathcal{B} \subset \mathbb{N}$. Each HYS is equipped with several sensors M_i^b and actuators G_j^b, with $i \in [1,m]$ and $j \in [1,n]$, where m and n are respectively the total number of measurement points and actuators, as shown in Figure 1.a. Upper indexes are omitted when not necessary for computation and comprehension. The structure of a hydrographic network is described by distinguishing the confluences (*see* Figure 1.a) and the difluences (*see* Figure 1.c).

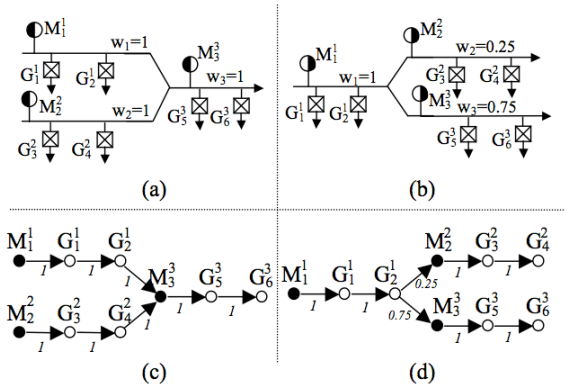


Figure 1: (a) A confluence, (c) its associated weighted digraph, (b) a difluence, (d) its associated weighted digraph.

According to the hydraulic conditions and the equations of energy and mass conservation, the sum of the discharges entering a node (confluent or difluent), is equal to the sum of the discharges outgoing from this node. Thus, around an operating point, the discharge q^b of the HYS b resulting from the confluence between several HYS is equal to the sum of the upstream HYS discharges, $q^b = \sum_{r \in \mathcal{C}^b} q^r$, where $\mathcal{C}^b \subset \mathcal{B}$ is the set of the HYS indices upstream to the HYS b. In addition, the HYS r resulting from the difluence of the HYS b upstream is supplied with a proportion w_r such that the discharge q^r verifies the relation: $q^r = w_r q^b$. In order to represent difluence, each HYS of an hydrographic system is associated to a discharge proportion w_r. For the HYS *source* and for the HYS downstream from a confluence (*see* Figure 1.a) it is equal to 1. The discharge proportion w_r of the HYS downstream the HYS b are known and such as $\forall r \in \mathcal{D}^b$, $w_r < 1$, and $\sum_{r \in \mathcal{D}^b} w_r = 1$, where $\mathcal{D}^b \subset \mathcal{B}$ is the set of HYS indices resulting from the difluence of the HYS b (*see* Figure 1.b). A discharge which is measured in a place of the hydrographic network, supplies the HYS downstream with discharge proportions according to the structure of the hydrographic network.

Table 1: Assignment function of **R** matrix.

```
Input: weighted digraph.
Output: proportion matrix R.
Initialization of R to 0
For each node h
  If h is a measurement point
     Run (h, h, 1,R)
  EndIf
EndFor

Run (h, c, p, R),
  For each successor d of c
    p_d ← p.w_d,
    Run (h, d, p_d, R),
    If d is a gate
       R(h,d) ← R(h,d) + p_d
    EndIf
  EndFor
```

The hydrographic systems are represented by a weighted digraph of instrumented points in order to determine the discharge proportions between two places of the networks. The digraph is composed with a succession of two types of nodes M_i or G_j, represented respectively by full circle and circle and their respective graphs, and arcs indicate the links between the successive nodes (*see* Figure 1.c and Figure 1.d). The arcs are oriented in the direction of the flow and are weighted by the discharge proportion between the two nodes w_r. Thereafter, an algorithm lead to the generation of the proportion matrix **R** which is composed of m lines (measurement points) and of n columns (actuators). The weighted digraph is browsed for each measurement point M_i following the algorithm given in Table 1. The matrix **R** contains all the discharge proportions of a point to another of the hydrographic networks.

Thereafter, the transfer time delay between the measurement points and the gates, is computed according to the method described in the next section.

3 IDENTIFICATION OF TRANSFER TIME DELAYS

Hydrographic systems consist of several reaches, *i.e.* a part between two measurement points, each reach being composed of Open-Channel Reach Section (OCRS), *i.e.* a part between two gates, between a measurement point and a gate or between a gate and a measurement point. The OCRS dynamics can be modelled by transfer functions according to the modelling method which consists in the simplification of the Saint Venant equations and their linearization around an operating point (Litrico and Georges, 1999; Malaterre and Baume, 1998; Chow et al., 1988). The parameters of the transfer function are considered constant under an operating range around the operating point. In this paper, only disturbances around the operating point are considered. Thus, the variation of the transfer delays for these discharges is sufficiently small in comparison with the chosen control period, and will not have a significant influence on the strategy effectiveness. If large operating conditions are considered, and/or in the case of the "small" control period, it is necessary to consider several time delays function of discharge value, as proposed in (Duviella et al., 2006). For each OCRS (*see* Figure 2), the transfer time delay τ_r is obtained from the step response of the corresponding transfer function. It is chosen as the time value for which Π_Q percent of step is reached. The percentage Π_Q can be tuned from simulation.

In the case of gridded systems, the value of the transfer time delay between the measurement point M_i^b and the gate G_j^d depends on the path to go from the measurement point M_i^b to the gate G_j^d (*see* Figure 2). $\mathcal{P}_{b,d}$ is the set of direct paths to go from the HYS b to the HYS d, and $P_v^{b,d}$ is one of the direct paths to go from the HYS b to the HYS d, such as $P_v^{b,d} \in \mathcal{P}_{b,d}$, where $1 \leq v \leq \rho_{b,d}$, with $\rho_{b,d}$ the total number of paths which compose $\mathcal{P}_{b,d}$. A direct path from M_i to G_j, is a path where not other measurement point can be met between M_i and G_j.

The transfer time delays between the measurement point M_i^b and the gate G_j^d are computed by considering each path and constitute the vector $\mathbf{t}_{M_i,j}$ ($\rho_{b,d} \times 1$):

$$\mathbf{t}_{M_i,j} = \left[t^1_{M_i,j}, t^2_{M_i,j}, \ldots, t^{\rho_{b,d}}_{M_i,j} \right]^T. \tag{1}$$

Thereafter, the transfer time delay between M_i^b and G_j^d, is computed according to the selected path $P_v^{b,d}$:

$$\begin{cases} t^v_{M_i,j} = t_{M_i,n_i} + \sum_{r=n_i}^{r<j} \tau^v_{r,r+1}, \\ n_i \leq j \leq n, \end{cases} \tag{2}$$

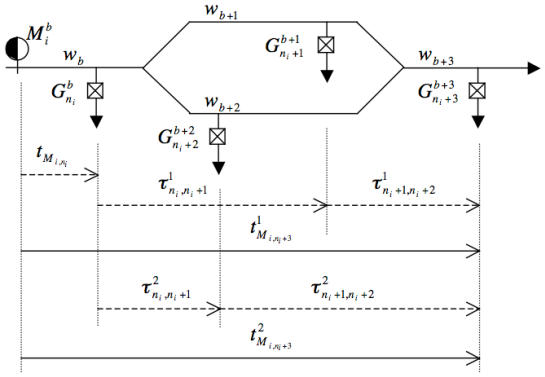


Figure 2: Transfer delays between the measurement point M_i and gates Gj.

where n_i is the first gate downstream M_i, v is the index of the path $P_v^{b,d}$, and $\tau^v_{r,r+1}$is the transfer time delay between each gate along the path $P_v^{b,d}$ as illustrated in Figure 2.

Then, the new setpoints must be assigned to the gates at a time instant taking into account the transfer time delays which are expressed according to the sampling period T_s:

$$kd^v_{M_i,j} = \left\lfloor \frac{t^v_{M_i,j}}{T_s} \right\rfloor + 1, \tag{3}$$

where $\lfloor x \rfloor$ denotes the integer part of x.

The measured water quantity in M_i, following the path with index v, will arrive on gate G_j at the time:

$$T^v_{M_i,j} = \left(k + kd^v_{M_i,j}\right) T_s. \tag{4}$$

Finally, the transfer time delays between the measurement point M_i^b and the gate G_j^d are expressed by the vector $\mathbf{T}_{M_i,j}$ ($\rho_{b,d} \times 1$):

$$\mathbf{T}_{M_i,j} = \left[T^1_{M_i,j}, T^2_{M_i,j}, \ldots, T^{\rho_{b,d}}_{M_i,j} \right]^T. \tag{5}$$

The complex hydrographic network representation, as well as the identification of the transfer time delays, constitute an essential step for the design of reactive control strategies.

4 SUPERVISION AND RESOURCE ALLOCATION

Supervision and hybrid control accommodation framework is depicted in Figure 3. The hydrographic network is represented by a set of m measurement points M_i and n gates G_j locally controlled. For each gate G_j, a weekly objective discharge $q_{j_{obj}}$ and seasonal weights λ_j and μ_j are given by the Management Objective Generation module according to the water

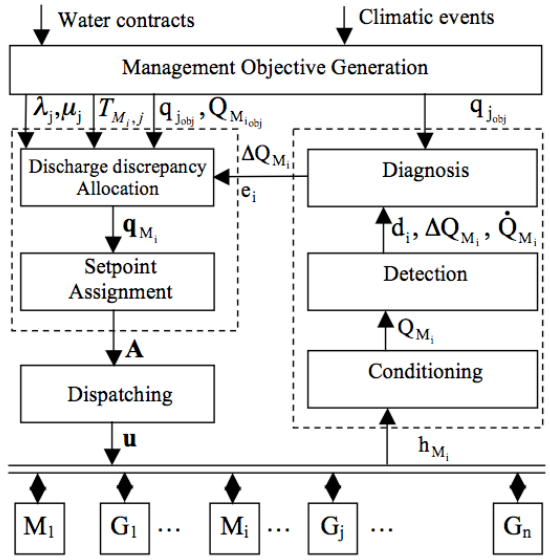


Figure 3: Supervision and hybrid control accommodation framework.

contracts and climatic events. The weekly measurement point objective discharge $Q_{M_{i_{obj}}}$ is known.

For each measurement point M_i ; $i = 1,...,m$, discharge supervision consists in monitoring discharge disturbances and diagnosing the resource state, simultaneously. Limnimeter measurements are conditioned by a low-pass filter on a sliding window which removes wrong data due to transmission errors for instance. Based on the discharge value Q_{M_i} which is determined at each sample time kT_s, detection and diagnosis automata are used respectively to detect a discharge discrepancy superior or inferior than a detection threshold d_{th} around $Q_{M_{i_{obj}}}$, and to diagnose the resource states (Duviella et al., 2007). According to the resource state and the discharge discrepancy $\Delta Q_{M_i} = Q_{M_{i_{obj}}} - Q_{M_i}$, the hybrid control accommodation consists in determining the setpoints q_j, and in assigning them to the gates taking into account the hydraulic system dynamics. The resource allocation consists in recalculating setpoints with a goal to route resource in excess to dams and to dispatch amongst the users the resource in lack. At each sample time kT_s, the resource allocation leads to the determination of allocation vector $\mathbf{q}_{M_i}$ which is composed of the new computed setpoints. The allocation vector is computed according to the resource state e_i tacking into account the seasonal weights λ_j and μ_j.

If the resource state is no diagnose situation, the setpoints are the objective discharges $q_{j_{obj}}$. The allocation vector is such as:

$$\mathbf{q}_{M_i} = \left[\delta^1_{\lceil R(i,1)\rceil} q_{1_{obj}} \ldots \delta^1_{\lceil R(i,j)\rceil} q_{j_{obj}} \ldots \delta^1_{\lceil R(i,n)\rceil} q_{n_{obj}}\right]^T, \quad (6)$$

where $\lceil \mathbf{x} \rceil$ corresponds to the higher rounding of x, n is the total number of gates, and δ^a_b the Kronecker index, is equal to 1 when $a = b$, and equal to 0 if not.

If the resource state is such as discharge is in lack or in excess**,** the water resource is allocated among the gate downstream the measurement point M_i, according to the weights λ_j and μ_j. The allocation strategy consists in optimizing a cost function by linear programming method for each measurement point. The cost function f_{M_i} is defined as the weighted sum of the differences between the setpoint q_j and the objective $q_{j_{obj}}$ for each gate G_j, at time kT_s:

$$f_{M_i} = \sum_{j=1}^{n} \left(\delta^1_{\lceil R(i,j)\rceil} \chi_{M_{i,j}} (q_j - q_{j_{obj}})\right), \quad (7)$$

with $\chi_{M_{i,j}} = \gamma \frac{1}{\lambda_j} + (\gamma - 1)\frac{1}{\mu_j}, \gamma = \frac{1}{2}(sign(\Delta Q_{M_i}) + 1)$.
The optimization is carried out under constraints:

$$\begin{cases} \sum_{j=1}^{n} \left(R(i,j)(q_j - q_{j_{obj}})\right) = \Delta Q_{M_i}, \\ q_{j_{\min}} \leq q_j \leq q_{j_{\max}}, \end{cases} \quad (8)$$

where $q_{j_{\min}}$ and $q_{j_{\max}}$ are respectively the minimum and maximum discharges given by gate, river or canal characteristics. In this case, the allocation vector $\mathbf{q}_{M_i}$ is such as:

$$\mathbf{q}_{M_i} = \left[\delta^1_{\lceil R(i,1)\rceil} \cdot q_1 \ldots \delta^1_{\lceil R(i,j)\rceil} \cdot q_j \ldots \delta^1_{\lceil R(i,n)\rceil} \cdot q_n\right]^T. \quad (9)$$

Then, to synchronize the gate control with the water lacks or excess due to the disturbances, the setpoints must be assigned at a time instant tacking into account the transfer time delays $T_{M_i,j}$ between the measurement point M_i and the gate G_j.

5 SETPOINT ASSIGNMENT RULES

The setpoint assignment consists in taking into account the transfer delays before the dispatching of the new computed setpoints at the gates. In the case of gridded systems, two different setpoint assignment rules are proposed.

The first rule consists in considering only one transfer delay $T_{M_i,j}$ from each measurement point M_i to each gate G_j, whatever existing several paths to go from M_i at the gate G_j. The transfer delay between M_i^b and G_j^d is selected as the direct path between M_i^b and G_j^d, which have the greatest supplying discharge proportion. The following assumptions are considered:

Table 2: Assignment function of α and β matrices.

```
Input: weighted digraph.
Output: α_{M_i} matrix, β_{M_i} matrices
Initialization of the diagonal of α_{M_i} to 0
Initialization of β_{M_i} to 0
g ← first gate successor of M_i
Run (M_i, g, 1, α_{M_i}, β_{M_i})

Run (M_i, c, p, α_{M_i}, β_{M_i})
  For any successor d of c
    p_d ← p.w_d
    If d is a gate
      Run (M_i, d, p_d, α_{M_i}, β_{M_i})
      α_{M_i}(d,d) ← α_{M_i}(d,d) + p_d
      l = 1
      While (β_{M_i}(l,d) ≠ 0)
        l++
      EndWhile
      β_{M_i}(l,d) ← p_d
    EndIf
  EndFor
```

- if the discharge proportion $\beta_{M_i}(v,j)$ resulting from M_i^b and supplying G_j^d by a single path $P_v^{b,d}$, is weak, the discrepancy allocation will be weak also,
- if the discharge proportion $\beta_{M_i}(v,j)$ resulting from M_i^b and supplying G_j^d by a single path $P_v^{b,d}$, is important, the discrepancy allocation will be important also.

The supplying discharge proportion β_{M_i} ($\rho_{M_i} \times n$), where ρ_{M_i} is the maximum number of paths between M_i and the gates G_j, is computed for each measurement point M_i according to the algorithm given in Table 2 and the weighted digraph of the system.

Thus, the set of allocation dates starting from M_i is denoted $\mathcal{T}_{M_i}$ ($1 \times n$) updated at each sampling period T_s and expressed by:

$$\mathcal{T}_{M_i} = [T_{M_i,1} \dots T_{M_i,j} \dots T_{M_i,n}], \tag{10}$$

where

$$\begin{cases} T_{M_i,j} = 0, & \text{if } \beta_{M_i}(1,j) = 0 \\ T_{M_i,j} = T^v_{M_i,j}, & \text{otherwise,} \end{cases} \tag{11}$$

and v such as $\beta_{M_i}(v,j) = \max_{l \in [1,\rho_{M_i}]} \beta_{M_i}(l,j)$. When $\beta_{M_i}(1,j) = 0$ there is no direct path between M_i and G_j.

At each sample time kT_s, the setpoint assignment matrix $\mathbf{A}^k_{M_i}$ ($H_{M_i} \times n$), where H_{M_i} is the allocation horizon from M_i, is scheduled according to $\mathcal{T}_{M_i}$ and $\mathbf{q}_{M_i}$. The first row of $\mathbf{A}^k_{M_i}$ contains the setpoints to be assigned to each gate from M_i at the date $(k+1)T_s$, the h^{th} row the ones to be assigned at the date $(k+h)T_s$ as defined in equation 12, the last row the ones to be assigned at the date $(k+H_{M_i})T_s$.

$$\begin{array}{l} \text{If } \mathcal{T}_{M_i}(j) \geq (k+h)T_s \\ \quad A^k_{M_i}(h,j) = q_{M_i}(j), \\ \text{Else} \\ \quad \text{If } 1 \leq h < H_{M_i} \\ \quad\quad A^k_{M_i}(h,j) = A^{k-1}_{M_i}(h+1,j) \\ \quad \text{Else} \\ \quad\quad A^k_{M_i}(h,j) = q_{j_{obj}} \\ \quad \text{Endif} \\ \text{Endif} \end{array} \tag{12}$$

and $A^0_{M_i}(h,j) = q_{j_{obj}}$.

The setpoints are dispatched with the control period $T_c = \kappa T_s$, where κ is an integer. The control setpoint vector denoted $\mathbf{u}$ ($1 \times n$) is updated at each date $k'T_c$, where $k' = \frac{k}{\kappa}$, thanks to the assignment matrix $\mathbf{A}^{k'}_{M_i}$ and the α_{M_i} ($n \times n$) diagonal control accommodation matrix, with $H = \frac{1}{\kappa} \max_{1 \leq i \leq m} (H_{M_i})$ the control horizon. For each measurement point M_i, the α_{M_i} matrix, the role of which is to capture the measurement point influence on the gates, must be determined. In order to generate the α_{M_i} matrix, the weighted digraph (*see* Figure 1.*c* and 1.*d*) is browsed using the algorithm given in Table 2, for each measurement point M_i. The control setpoint vector $\mathbf{u}^{k'}$ ($1 \times n$) is calculated by:

$$u^{k'}(j) = \sum_{i=1}^{m} \alpha_{M_i}(j,j) A^{k'}_{M_i}(1,j). \tag{13}$$

The second rule consists in considering the several direct transfer delays $\mathbf{T}_{M_i,j}$ from each measurement point M_i to each gate G_j. The set of allocation dates starting from M_i is denoted $\mathcal{T}_{M_i}$ ($\rho_M \times n$), where ρ_M is the maximum number of paths between the measurement points M_i and the gates G_j. The matrix $\mathcal{T}_{M_i}$ is updated at each sampling period T_s and expressed by:

$$\mathcal{T}_{M_i} = [\mathbf{T}_{M_i,1} \dots \mathbf{T}_{M_i,j} \dots \mathbf{T}_{M_i,n}], \tag{14}$$

where $\mathbf{T}_{M_i,j}$ is the set of the transfer time delays between the measurement point M_i and the gate G_j. The value of $T_{M_i,j}(l)$ is fixed to 0 for $\rho_{M_i} < l \leq \rho_M$, *i.e.* when the length ρ_{M_i} of the $\mathbf{T}_{M_i,j}$ is smaller than ρ_M.

In this case, the setpoint assignment matrix $\mathbf{A}^k_{M_i}$ ($H_{M_i} \times n$) is scheduled, at each sample time kT_s, ac-

cording to $\mathcal{T}_{M_i}$ and $\mathbf{q}_{M_i}$:

$$\begin{array}{l}\text{If } \exists v \text{ such as } \mathcal{T}_{M_i}(v,j) \geq (k+h)T_s \\ \quad A^k_{M_i}(h,j) = \sum_{v=1}^{\rho_M} \varphi_v.\beta_{M_i}(v,j).q_{M_i}(j) \\ \text{Else} \\ \quad \text{If } 1 \leq h < H_{M_i} \\ \quad\quad A^k_{M_i}(h,j) = A^{k-1}_{M_i}(h+1,j) \\ \quad \text{Else} \\ \quad\quad A^k_{M_i}(h,j) = \alpha_{M_i}(j,j).q_{j_{obj}} \\ \quad \text{Endif} \\ \text{Endif}\end{array} \tag{15}$$

where $\varphi_v = 1$ if $\mathcal{T}_{M_i}(v,j) \geq (k+h)T_s$, $\varphi_v = 0$ otherwise, and $A^0_{M_i}(h,j) = \alpha_{M_i}(j,j).q_{j_{obj}}$.

The control setpoint vector denoted $\mathbf{u}$ $(1 \times n)$ is updated at each date $k'T_c$, thanks to the assignment matrix $\mathbf{A}^{k'}_{M_i}$, with $H = \frac{1}{\kappa}\max_{1\leq i\leq m}(H_{M_i})$ the control horizon. The control setpoint vector $\mathbf{u}^{k'}$ $(1 \times n)$ is calculated by:

$$u^{k'}(j) = \sum_{i=1}^{m} A^{k'}_{M_i}(1,j). \tag{16}$$

The setpoint dispatching leads to the application of the most recently calculated setpoints. This method increases the control strategy reactivity, because discharge variations between two control dates are taken into account.

6 SIMULATION RESULTS

The proposed setpoints assignment rules have been evaluated for a hydrographic system composed of one difluence and one confluence (*see* Figure 4).

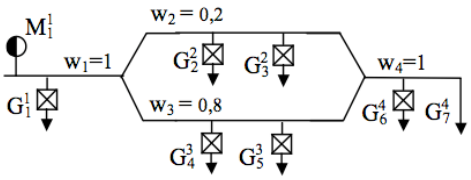


Figure 4: Hydrographic system composed of one difluence and one confluence.

The hydrographic system is composed of 4 HYS, equipped with 6 gates, G^1_1 to G^4_6, and 1 measurement point M^1_1. The discharge downstream the gate G^1_1 fed the HYS which is equipped with the gates G^2_2 to G^2_3 with the discharge proportion w_2 and the HYS which is equipped with the gates G^3_4 to G^3_5 with the discharge proportion w_3. The discharge proportion w_2 is equal to 0.2 and w_3 to 0.8. The gates G^4_7 which corresponds to the canal outputs, is not controlled. The gate characteristics, *i.e.* objective discharge $q_{j_{obj}}$, maximum and minimum discharges $q_{j\max}$, $q_{j\min}$, and their associated weights, are given in Table 3.

Table 3: Gate parameters.

Gate	$q_{j_{obj}}$ $[m^3/s]$	$q_{j\min}$ $[m^3/s]$	$q_{j\max}$ $[m^3/s]$	λ_j	μ_j
G^1_1	1.1	0.05	0.85	10	10
G^2_2	0.3	0.1	0.9	1	4
G^2_3	0.4	0.15	1.2	1	4
G^3_4	1.9	0.1	1.4	4	1
G^3_5	1.6	0.1	0.9	1	4
G^4_6	0.9	0.05	1.8	10	10
G^4_7	1.8	0.05	0.75	–	–

The use of the proposed rules requires the identification of the transfer time delays. The set of HYS which are characterized by trapezoidal profile have been modelled according to the transfer time delay identification steps. The matrix, $\mathbf{T}_{M_i}$, of transfer time delays between M_1 and each gate, expressed in seconds, are given by:

$$\mathbf{T}_{M_i} = \begin{bmatrix} 850 & 1750 & 2700 & 1450 & 2050 & 3750 \\ 0 & 0 & 0 & 0 & 0 & 2700 \end{bmatrix}. \tag{17}$$

There are two identified transfer time delays between M_1 and G_6; $T^1_{M_1,6} = 3750\ s$ corresponds to the path $P^{1,4}_1$; $T^2_{M_1,6} = 2700\ s$ corresponds to the path $P^{1,4}_2$.

Then, the hydrographic system (*see* Figure 4) is represented by the weighted digraph depicted in Figure 5 to determine the matrix $\mathbf{R}$, and then, to determine the matrices α_{Mi} and β_{M_i}.

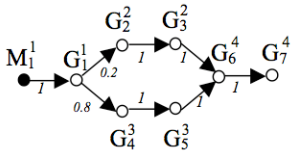


Figure 5: Graph for the determination of $\mathbf{R}$ and α_{Mi}.

The matrix $\mathbf{R}$ is given by:

$$\mathbf{R} = \begin{bmatrix} 1 & 0.2 & 0.2 & 0.8 & 0.8 & 1 \end{bmatrix}. \tag{18}$$

The diagonal matrix α_{M_1} is given by:

$$\alpha_{M_1} = diag\{1, 0.2, 0.2, 0.8, 0.8, 1\}. \tag{19}$$

The matrix β_{M_1} is given by:

$$\beta_{M_1} = \begin{bmatrix} 1 & 0.2 & 0.2 & 0.8 & 0.8 & 0.2 \\ 0 & 0 & 0 & 0 & 0 & 0.8 \end{bmatrix}. \tag{20}$$

The objective discharges of M_1 correspond to 8 m^3/s. The hydrographic system is subjected to disturbances upstream the measurement points M_1 (*see*

Figure 6.*a*). The detection threshold is selected as d_{th}= 0.15 m^3/s. Figure 6 shows discharges measured on M_1, and the new setpoints which have been dispatched at the gates which were controlled, *i.e.* G_1 in (b) and G_6 in (c), and the discharges resulting at the canal ends q_7 in (d) in case 1: the case where only one transfer time delay is considered (the first rule is used without any assumption about the discharge proportion values), the transfer time delay considered is $T^1_{M_1,6}$ (dashed line), in case 2: the case where the first rule is applied, thus the time delay considered is $T^2_{M_1,6}$ (dotted line), and in case 3: the case where the second rule is applied (continuous line).

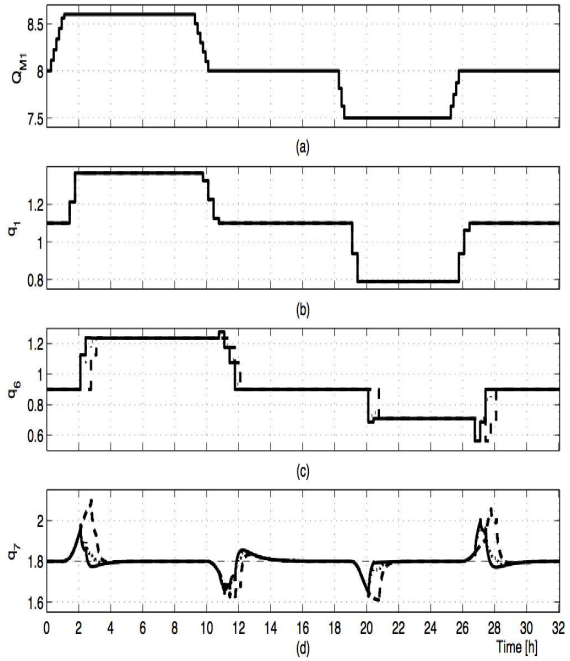


Figure 6: Discharges in $[m^3/s]$ (a) Q_{M_1}, (b) q_1, (c) q_6, and (d) the resulting discharges q_7.

Whatever the setpoint assignment rules used are, there is a peak of approximately 0.15 m^3/s on G_7 at the 2^{nd}, 11^{th}, 20^{th} and 27^{th} hours (*see* Figure 6.*d*), due to the detection threshold d_{th} and the occurrence of the discharge discrepancies on Q_{M_1}. When the transfer time delay is $T^1_{M_1,6}$, the setpoints are assigned too late, and the discharges at the end of the hydrographic system are not close to the objective value q_{7obj}. These results show the importance of the transfer time delay. The results are improved when the first rule is used with $T^2_{M_1,6}$, because of better evaluation of transfer time delay. Finally, the performances are also improved when the second rule is applied, the effective transfer time delays are taken into account because all direct paths are considered. The maximum and minimum discharges reached at G_7 and the water volume V which was not allocated are displayed in Table 4. The maximum discharge discrepancy at G_7 corresponds to 9.5 % of the objective discharge q_{7obj} when the second rule is used and to 10.5 % in the other case. The second rule leads to spare an additional water quantity of 42 m^3 during 32 hours, in comparison to the use of the first rule. The differences between the two strategies are weak. In addition, these differences decrease for hydrographic systems which are equipped by a great number of measurement points, because, in this case, the number of direct paths is weak.

Table 4: Criteria computed when the different rules are used.

Case	$\max(q_7)$ $[m^3/s]$	$\min(q_7)$ $[m^3/s]$	V $[m^3]$
case 1	2.09	1.59	1815
case 2	1.99	1.63	1099
case 3	1.97	1.63	1057

7 CONCLUSION

The resource allocation and setpoint assignment rules constitute a generic approach allowing the water resource valorization whatever the configuration of the hydrographic networks is. Multiple graph representations make it possible to identify the information for implementing the proposed supervision and hybrid control accommodation strategy. Two rules of setpoint assignment have been proposed, tested and compared within the framework of a networked open-channel system composed of one difluent and one confluent. Although the second rule leads to the best performances, its implementation is more complex than the one for the first rule. The choice between the two strategies could be carried out only by considering the hydrographic system with this equipment.

REFERENCES

Cembrano, G., Wells, G., Quevedo, J., Perez, R., and Argelaguet, R. (2000). Optimal control of a water distribution network in a supervisory control system. *Control Engineering Practice*, 8:1177–1188.

Chan, C., Kritpiphat, W., and Tontiwachwuthikul, P. (1999). Development of an intelligent control system for a municipal water distribution network. *IEEE Canadian conference on Electrical and Computer Engineering*, 2:1108–1113.

Chow, V. T., Maidment, D. R., and Mays, L. W. (1988). *Applied Hydrology*. McGraw-Hill, New York, Paris.

Duviella, E., Chiron, P., and Charbonnaud, P. (2006). Hybrid control accommodation for water-asset management of hydraulic systems subjected to large operating conditions. *ALSIS06, 1st IFAC Workshop on Applications of Large Scale Industrial Systems, Helsinki, Finland, August 30-31 2006*.

Duviella, E., Chiron, P., Charbonnaud, P., and Hurand, P. (2007). Supervision and hybrid control accommodation for water asset management. *Control Engineering Practice (CEP)*, 15:17–27.

Faye, R. M., Sawadogo, S., Niang, A., and Mora-Camino, F. (1998). An intelligent decision support system for irrigation system management. *IEEE International Conference on Systems, Man and Cybernetics, SMC'98, October 11-14, San Diego, USA*, 4:3908–3913.

Islam, A., Raghuwanshi, N. S., Singh, R., and Sen, D. J. (2005). Comparison of gradually varied flow computation algorithms for open-channel network. *Journal of irrigation and drainage engineering*, 131(5):457–465.

Lisounkin, A., Sabov, A., and Schreck, G. (2004). Interpreter based model check for distribution networks. *IEEE international conference on industrial informatics, INDIN'O4, 24-26 juin*, pages 431–435.

Litrico, X. and Georges, D. (1999). Robust continuous-time and discrete-time flow control of a dam-river system. (i) modelling. *Applied Mathematical Modelling 23*, pages 809–827.

Malaterre, P.-O. and Baume, J.-P. (1998). Modeling and regulation of irrigation canals: Existing applications and ongoing researches. *IEEE International Conference on Systems, Man, and Cybernetics*, 4:3850–3855.

Mansour, H. E. F., Georges, D., and Bornard, G. (1998). Optimal control of complex irrigation systems via decomposition-coordination and the use of augmented lagrangian. *IEEE, International Conference on Control Applications, Trieste, Italy*, pages 3874–3879.

Naidu, B. J., Bhallamudi, S. M., and Narasimhan, S. (1997). GVF computation in tree-type channel networks. *Journal of hydraulic engineering*, 123(8):700–708.

BOUNDARY CONTROL OF A CHANNEL
Last Improvements

Valérie Dos Santos
Université de Lyon, Lyon, F-69003, France ; Université Lyon 1, CNRS, UMR 5007, LAGEP, Villeurbanne, F-69622, France
ESCPE, Villeurbanne, F-69622, France
dossantos@lagep.cpe.fr

Christophe Prieur
LAAS-CNRS, 7 avenue du Colonel Roche, 31077 Toulouse, Cedex 4, France
christophe.prieur@laas.fr

Keywords: Lyanpunov stability, Proportionnal-Integral control, Saint-Venant equations, Riemann coordinates.

Abstract: Different improvements have been developed in regards to the stability and the control of two-by-two non linear systems of conservation laws, and in particular for the Saint-Venant equations and the control of flow and water level on irrigation channel. One stability result based on the Riemann coordinates is presented here and sufficient conditions are given to insure the Cauchy convergence. Another result still based on the Riemann approach is presented too, in the linear case, to improve the feedback control based on the Riemann invariants.

1 INTRODUCTION

In this paper, we are concerned with the stability of the non linear Saint-Venant equations, a two-by-two systems of conservation laws, that are described by hyperbolic partial differential equations, with one independent time variable $t \in [0,\infty)$ and one independent space variable, $x \in [0,L]$. For such systems, the considered boundary control problem is the problem of designing feedback control actions at the boundaries (i.e. at $x = 0$ and $x = L$) in order to ensure that the smooth solution of the Cauchy problem converge to a desired steady state.
This problem has been previously considered in the literature ((Litrico et al., 2005)), and in our previous papers (Coron et al., 2002). Those results have been improved in (Dos Santos et al., 2007) in order to take account of non homogeneous terms (like perturbations, slope or frictions) adding an integral part to the Riemann control developed.
Recently, the non linear problem of the stability of systems of two conservation laws perturbed by non homogeneous terms has been investigated (Prieur et al., 2006), (Dos Santos and Prieur, 2007), using the state evolution of the Riemann coordinates.
This paper aim is to shortly present both last results develop on (Dos Santos and Prieur, 2007), (Dos Santos et al., 2007) and to illustrate them with simulations and experimentations based on a river data and the Valence micro-channel respectively.
After a short presentation of the shallow water equations, the first problem is stated, the tools presented, and the stability result established. The second result is developed in the same way in the fourth section and the simulations results are produced as well as the experimentations ones in the last part.

2 DESCRIPTION OF THE MODEL: SAINT-VENANT EQUATIONS

We consider a reach of an open channel as represented in Figure 1.
We assume that the channel is prismatic with a constant rectangular section. Note that in our configuration, the slope could be non null as well as the friction effects.
The flow dynamics are described by a system of two laws of conservation (Saint-Venant or shallow water equations), namely a law of mass conservation and a law of momentum conservation

$$\partial_t H + \partial_x (Q/B) = 0, \qquad (1)$$

$$\partial_t Q + \partial_x \left(\frac{Q^2}{BH} + \frac{1}{2} g B H^2\right) = gBH(I - J), \qquad (2)$$

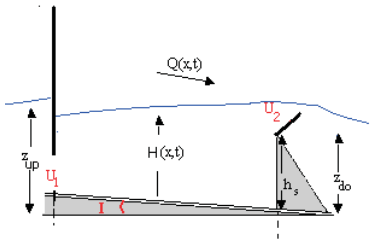


Figure 1: Scheme of a channel: one reach with an overflow gate.

where $H(t,x)$ stands for the water level and $Q(t,x)$ the water flows in the reach while g denotes the gravitation constant ($m.s^{-2}$). I is the bottom slope ($m.m^{-1}$), B is the channel width (m) and J is the slope's friction ($m.m^{-1}$).
The slope's friction J is expressed with the Manning-Strickler expression, (n_M is the Manning coefficient ($s.m^{-1/3}$) and the Strickler coefficient is $K = \frac{1}{n_M}$ ($m^{1/3}.s^{-1}$)),

$$J(Q,H) = \frac{n_M^2 Q^2}{S(H)^2 R(H)^{4/3}},$$

where $S(H)$ is the wet surface (m^2) and $P(H)$ the wet perimeter (m): $S(H) = BH, P(H) = B+2H$, $R(H)$ is the hydraulic radius (m), $R(H) = S(H)/P(H)$.

The control actions are the positions U_0 and U_L of the two spillways located at the extremities of the pool and related to the state variables H and Q by the following expressions.
Two cases may occur for the gate equations at $x = 0$ and $x = L$:

- a submerged underflow gate:

$$Q(x_i,t) = U_i B \mu_i \sqrt{2g(H_1(x_i,t) - H_2(x_i,t))} \quad (3)$$

- or a submerged overflow gate:

$$H_1(x_i,t) = \left(\frac{Q^2(x_i,t)}{2gB^2\mu_i^2}\right)^{1/3} + h_{s,i} + U_i, \quad (4)$$

where $H_1(x,t)$ is the water level before the gate, $H_2(x,t)$ is the water level after the gate and $h_{s,i}$ is the height of the fixed part of the overflow gate n°i (Fig. (1)) and μ_i is the water flow coefficient of the gate n°i located at $x = x_i$.
Note that the system (1)-(2) is strictly hyperbolic, i.e. its Jacobian matrix has two non-zero real distinct eigenvalues:

$$\lambda_1(H,V) = \frac{Q}{BH} + \sqrt{gH},\ \lambda_2(H,V) = \frac{Q}{BH} - \sqrt{gH}.$$

They are generally called *characteristic velocities*.
The flow is said to be *fluvial* (or subcritical) when the characteristic velocities have opposite signs:

$$\lambda_2(H,V) < 0 < \lambda_1(H,V).$$

Different stability results have been given for the linearized system (Coron et al., 2007)-(Dos Santos et al., 2007) and the non-linear one (Prieur et al., 2006)-(Dos Santos and Prieur, 2007) using the properties of Riemann coordinates. Those results are quickly resumed in both following sections.

3 FIRST RESULT: INTEGRAL ACTIONS AND LYAPUNOV STABILITY ANALYSIS

3.1 Linearized System

An equilibrium (H_e, Q_e) is a constant solution of the equations (1)-(2) , i.e. $H(t,x) = H_e$, $Q(t,x) = Q_e\ \forall t$ and $\forall x$ which satisfies the relation:

$$J(H_e, Q_e) = I. \quad (5)$$

A linearized model is used to describe the variations around this equilibrium. The following notations are introduced:

$$h(t,x) \hat{=} H(t,x) - H_e(x),\ q(t,x) \hat{=} Q(t,x) - Q_e(x).$$

The linearized model around the equilibrium (H_e, Q_e) is then written as

$$\partial_t h(t,x) + \partial_x q(t,x) = 0 \quad (6)$$

$$\partial_t q(t,x) + cd\partial_x h(t,x) + (c-d)\partial_x q(t,x) = -\gamma h(t,x) - \delta q(t,x), \quad (7)$$

with:

$$c = \sqrt{gBH_e} + \frac{Q_e}{H_e\sqrt{B}},\ d = \sqrt{gBH_e} - \frac{Q_e}{H_e\sqrt{B}}$$

$$\gamma = gBH_e \frac{\partial J}{\partial H}(H_e, Q_e),\ \delta = gBH_e \frac{\partial J}{\partial Q}(H_e, Q_e).$$

In the special case where the channel is horizontal ($I = 0$) and the friction slope is negligible ($n \approx 0$), we observe that $\gamma = \delta = 0$ and that this linearized system is exactly in the form of the following linear hyperbolic system:

$$\partial_t h(t,x) + \partial_x q(t,x) = 0 \quad (8)$$

$$\partial_t q(t,x) + cd\partial_x h(t,x) + (c-d)\partial_x q(t,x) = 0. \quad (9)$$

It is therefore legitimate to apply the control with integral actions that have been analyzed in (Coron et al., 2007) to open channels having small bottom and friction slopes.

3.2 Riemann Coordinates and Stability Conditions

In order to solve this boundary control problem, the *Riemann coordinates* (see e.g. (Renardy and Rogers, 1993) p. 79) defined by the following change of coordinates are introduced :

$$a(t,x) = q(t,x)+dh(t,x) \quad (10)$$
$$b(t,x) = q(t,x)-ch(t,x) \quad (11)$$

With these coordinates, the system (8)-(9) is written under the following diagonal form :

$$\partial_t a(t,x)+c\partial_x a(t,x)=0 \quad (12)$$
$$\partial_t b(t,x)-d\partial_x b(t,x)=0 \quad (13)$$

In the Riemann coordinates, the control problem can be restated as the problem of determining the control actions in such a way that the solutions $a(t,x)$, $b(t,x)$ converge towards zero.
The boundary control laws $u_0(t)$ and $u_L(t)$ are defined such that the boundary conditions (3)-(4) expressed in the Riemann coordinates satisfy the linear relations (Coron et al., 2007) augmented with appropriate integrals as follows:

$$a_0(t)+k_0b_0(t)+m_0y_0(t) = 0 \quad (14)$$
$$b_L(t)+k_La_L(t)+m_Ly_L(t) = 0 \quad (15)$$

where k_0, k_L and m_0, m_L are constant design parameters that have to be tuned to guarantee the stability. The integral y_0 on the flow q at the boundary $x=0$ and the integral y_L on the other state h at the boundary $x=L$ are defined as:

$$y_0(t) = \int_0^t q_0(s)ds = \int_0^t \frac{ca_0(s)+db_0(s)}{c+d}ds$$
$$y_L(t) = \int_0^t h_L(s)ds = \int_0^t \frac{a_L(s)-b_L(s)}{c+d}ds.$$

Using Lyapunov theory, one can prove this theorem:

Theorem 1 *Let m_0, m_L and k_0, k_L four constants such that the six following inequalities hold:*

$$m_0>0, \qquad m_L<0, \quad (16)$$
$$|k_0|<1, \qquad |k_0k_L|<1, \quad (17)$$
$$|k_L|<\frac{c}{d} \qquad \frac{d}{c}<1, \quad (18)$$

Then there exist five positive constants A, B, μ, N_0 and N_L such that, for every solution $(a(t,x),b(t,x))$, $t\geqslant 0$, $x\in[0,L]$, of (12), (13), (14) and (15) the following function:

$$U(t) = \frac{A}{c}\int_0^L a^2(t,x)e^{-\mu x/c}dx+\frac{c+d}{2}N_0y_0^2(t) + \frac{B}{d}\int_0^L b^2(t,x)e^{\mu x/d}dx+\frac{c+d}{2}N_Ly_L^2(t)$$

satisfies:

$$\dot{U}\leq -\mu U.$$

Remark 1 *As it has been mentioned above, in our previous paper (Coron et al., 2007) the special case with $m_0=m_L=0$ in the boundary conditions (14)-(15) and $N_0=0$, $N_L=0$ has been treated. We have shown that inequality $|k_0k_L|<1$ is sufficient to have $\dot{U}<-\mu U$ for some $\mu>0$ along the system trajectories and ensure the convergence of $a(t,x)$ and $b(t,x)$ to zero.*

In the fifth section, we shall illustrate the efficiency of the control with simulations on a realistic model of a waterway and with experimental results on a real life laboratory plant.

4 SECOND RESULT: STABILITY OF THE NON-LINEAR SAINT-VENANT EQUATIONS

Previous result delead with the stability of two conservation laws systems, which can be written as (8)-(9) (Coron et al., 2007), i.e. for homogeneous hyperbolic systems. The stability condition depends thus of the spectral radius of the Jacobian matrix linked. In (Prieur et al., 2006), those results have been extended to the non homogeneous system, with an additional condition on the size of the non homogeneous terms. Here, we proposed a new result that improve the sufficient stability condition $|k_0k_L|<1$ (Dos Santos and Prieur, 2007).

4.1 Statement

In order to introduce the problem under consideration in this work, we need some additional notations:

- The usual euclidian norm $|\cdot|$ in $\mathbb{R}$ is denoted by $|\cdot|$. The ball centered in $0\in\mathbb{R}$ with radius $\varepsilon>0$ is denoted $B(\varepsilon)$;
- Given Φ continuous on $[0,L]$ and Ψ continuously differentiable on $[0,L]$, we denote

$$|\Phi|_{C^0(0,L)} = \max_{x\in[0,L]}|\Phi(x)|,$$
$$|\Psi|_{C^1(0,L)} = |\Psi|_{C^0(0,L)}+|\Psi'|_{C^0(0,L)};$$

- the set of continuously differentiable functions $\Psi^{\#}$: $[0,L]\to\mathbb{R}$ satisfying the compatibility assumption $\mathcal{C}$ and $|\Psi^{\#}|_{C^1(0,L)}\leq\varepsilon$ is denoted $B_{\mathcal{C}}(\varepsilon)$.

For constant control actions $U_0(t)=\bar{U_0}$ and $U_L(t)=\bar{U_L}$, a *steady-state solution* is a constant solution $(H,Q)(t,x)=(\bar{H},\bar{Q})(x)$ for all $t\in[0,+\infty)$,

for all $x \in [0,L]$ which satisfies (1)-(2) and the boundary conditions (3)-(4).

At time $t \geq 0$, *the output* of the system (1)-(2) is given by the following

$$y(t) = (H_0(t), H_L(t)) \tag{19}$$

The problem under consideration in this work is the following: *Given a steady-state $\left(\bar{H},\bar{Q}\right)^T$, called the set point, we consider the problem of the local exponential stabilization of (1)-(2) by means of a boundary output feedback controller, i.e. we want to compute a boundary output feedback controller $y \mapsto (U_0(y), U_L(y))$ such that, for any smooth small enough (in C^1-norm) initial condition $H^\#$ and $Q^\#$ satisfying our compatibility conditions, the PDE (1)-(2) with the boundary conditions (3)-(4) and the initial condition*

$$(H,Q)(x,0) = (H^\#, Q^\#)(x) \ , \forall x \in [0,L]. \tag{20}$$

has a unique smooth solution converging exponentially fast (in C^1-norm) towards $\left(\bar{H},\bar{Q}\right)^T$.

4.2 Stability Result

First note that the system (1)-(2) is strictly hyperbolic, i.e. the Jacobian matrix of this system has two non-zero real distinct eigenvalues:

$$\lambda_1(H,Q) = \frac{Q}{BH} + \sqrt{gH},\ \lambda_2(H,Q) = \frac{Q}{BH} - \sqrt{gH}.$$

They are generally called *characteristic velocities*.

The flow is said to be *fluvial* (or subcritical) when the characteristic velocities have opposite signs:

$$\lambda_2(H,Q) < 0 < \lambda_1(H,Q).$$

Under constant boundary conditions $Q(0,t) = \bar{Q}_0$ and $H(L,t) = \bar{H}_L$, for all t, there exists a steady state solution $x \mapsto (\bar{Q},\bar{H})$ satisfying

$$\partial_x \bar{Q}(x) = 0, \qquad \partial_x \bar{H}(x) = -g\bar{H}\frac{I-\bar{J}}{\bar{\lambda}_1\bar{\lambda}_2}, \tag{21}$$

with $\bar{\lambda}_1 = \lambda_1(\bar{H},\bar{Q})$, and $\bar{\lambda}_2 = \lambda_2(\bar{H},\bar{Q})$.

Let t_1 and t_2 be the time instants defined by

$$x_1(t_1) = L \ , \ x_2(t_2) = 0, \tag{22}$$

where x_i, $i = 1,2$, are the solution of the Cauchy problem

$$\dot{x}_i(t) = \lambda_i(\bar{H},\bar{Q}),\ x_1(0) = 0,\ x_2(L) = 0.$$

To state our stability result, we need to introduce the following notations

$$\bar{a} = (\frac{\bar{Q}}{\bar{B}\bar{H}} + 2\sqrt{g\bar{H}}) \ , \ \bar{b} = (\frac{\bar{Q}}{\bar{B}\bar{H}} - 2\sqrt{g\bar{H}}).$$

We can explicit functions f_1 and f_2, and expressions ℓ_1 and ℓ_2 depending on the equilibrium and on the perturbations such that, for all $i \in \{1,2\}$,

$$\ell_i = f_i(\bar{\lambda}_i, \bar{a}, \bar{b}, I, n_M) \ . \tag{23}$$

Due to space limitation, the explicit expression of ℓ_1 and ℓ_2 is omitted, it is developed in (Dos Santos and Prieur, 2007).

The boundary conditions are written as follows:

$$a(t,0) + k_0 b(t,0) = 0 \tag{24}$$
$$b(t,L) + k_L a(t,L) = 0, \tag{25}$$

where k_0, k_L are constant design parameters that have to be tuned to guarantee the stability.

We are now in position to state our stability result, here in the case of a reach bounded by two underflow gates:

Theorem 2 *Let t_1, t_2, ℓ_1 and ℓ_2 be defined by (22), and (23) respectively.*

If the bottom slope function I, the slope's friction function J are sufficiently small in C^1 norm, then we have

$$\max(t_1\ell_1, t_2\ell_2) < 1 \ , \tag{26}$$

In that case, there exist k_0 and k_L such that

$$|\, k_0 k_L \,| + t_2 \,|\, k_0 \,|\, \ell_2 + t_1\ell_1 < 1, \tag{27}$$
$$|\, k_0 k_L \,| + t_1 \,|\, k_L \,|\, \ell_1 + t_2\ell_2 < 1. \tag{28}$$

The following boundary output feedback controller

$$U_0 = H_0 \frac{\frac{\bar{Q}_0}{BH_0} - 2\sqrt{g}\alpha_0\left(\sqrt{H_0} - \sqrt{\bar{H}_0}\right)}{\mu_0\sqrt{2g(z_{up} - H(0,t))}}, \tag{29}$$

$$U_L = H_L \frac{\frac{\bar{Q}_L}{BH_L} + 2\sqrt{g}\alpha_L\left(\sqrt{H_L} - \sqrt{\bar{H}_L}\right)}{\mu_L\sqrt{2g(H(L,t) - z_{do})}}, \tag{30}$$

where $H_0 = H(t,0)$, $H_L = H(t,L)$, $\alpha_0 = \frac{1-k_0}{1+k_0}$, and $\alpha_L = \frac{1-k_L}{1+k_L}$ make the closed loop system locally exponentially stable, i.e. there exist $\varepsilon_0 > 0$, $C > 0$ and $\mu > 0$ such that, for all initial conditions $(H^\#, Q^\#)$: $[0,L] \to (0,+\infty)$ continuously differentiable, satisfying some compatibility conditions and the inequality

$$|(H^\#, Q^\#) - (\bar{H},\bar{Q})|_{C^1(0,L)} \leq \varepsilon,$$

there exists a unique C^1 solution of the Saint-Venant equations (1)-(2), with the boundary conditions (3)-(4) and the initial condition (20), defined for all $(x,t) \in [0,L] \times [0,+\infty)$. Moreover it satisfies, $\forall t \geq 0$,

$$|(H,Q) - (\bar{H},\bar{Q})|_{C^1(0,L)} \leq C_1 e^{-\mu\, t} |(H^\#, Q^\#)|_{C^1(0,L)} \ .$$

This result is proved using Riemann coordinates formalism, the Saint-Venant equations are rewritten in Riemann coordinates. Due to the slope's friction J and the bottom slope I, it gives rise to a system of conservation laws with non-homogeneous terms. The evolution of the Riemann coordinates along the characteristic curves are estimated. This estimation could be possible as soon as the non-homogenous terms are sufficiently small. A sufficient condition in terms of the boundary conditions for the asymptotic stability of the Riemann coordinates is given. This necessary condition is written as (26) in terms of the variables H and Q.
This result is illustrated in the following part.

5 NUMERICAL SIMULATIONS AND EXPERIMENTS

In this section we applied both result on numerical simulations of a river and on an experimental setup. In both cases, the assumption (26) is satisfied, thus we succeed to design an stabilizing boundary output feedback controller. Let us note that if the inequalities (27) and (28) hold then we have

$$| k_0 k_L | < \min(1 - t_1 \ell_1, 1 - t_2 \ell_2). \qquad (31)$$

In the same way, conditions (16)-(18) are satisfied.

5.1 Simulation Results on a River

To illustrate our results, simulations have been realized with the realistic data of a river, on the software SIC developed by the CEMAGREF. Physical parameters of this river are given in Table 1, and the gates are overflow ones.

Table 1: Parameters of one reach of the river.

parameters	$B(m)$	$L(m)$	μ
values	3	2272	0.6
parameters	slope $I(m^1.s^{-1})$	K $(m^{1/3}.s^{-1})$	
values	$1.8046e^{-4}$	60	

One series of simulations is described (Fig. 2) the initial condition are the following:
$Q_e(0) = 2m^3.s^{-1}$, $Q_e(L) = 0.7m^3.s^{-1}$, $H_e(0) = 1.41m$, $z_e(L) = 1.8m$.
The steady state to reach is defined by:
$\bar{Q}(0) = 2m^3.s^{-1}$, $\bar{Q}(L) = 0.7m^3.s^{-1}$, $\bar{H}(0) = 1.85m$, $\bar{H}(L) = 2.26m$.

Using (31), we note that the tuning parameters should satisfy $k_0 k_L < k_0 k_{Lmax} = 0.1682$.

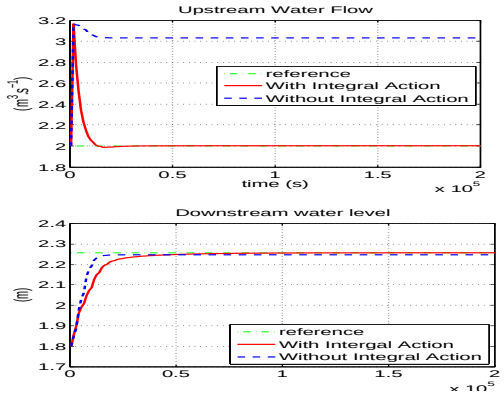


Figure 2: Water flow at upstream and level at downstream.

Two simulations are pictured, with the following values $k_0 k_L = 0.0039$, and

1. $m_0 = 0 = m_L$,
2. $m_0 = -0.0001$ $m_L = 0.001$.

Other simulations with higher values of $k_0 k_L$ ($k_0 k_L > k_0 k_{Lmax}$) diverge in the sense that the water flow and level do not converge to the steady state required or oscillate.
All the simulations shows the well suitability of the two stability tests (27)-(28) and of the condition(31), the **three** have to be verified to insure the stability of the system.
The stability hypothesis (16)-(18) linked to the integral actions are checked, even if it is applied to the non linear system.

5.2 Experimental Results on a Micro-channel

An experimental validation has been performed on the Valence micro-channel (Tab.2). This pilot channel is located in Valence (France). It is operated under the responsibility of the LCIS [1] laboratory. This experimental channel (total length=8 meters) has an adjustable slope and a rectangular cross-section (width=0.1 meter). The channel is ended at downstream by a variable overflow spillway and furnished with three underflow control gates (Fig. (3)).

Table 2: Parameters of the channel of Valence.

parameters	$B(m)$	$L\ (m)$	$K\ (m^{1/3}.s^{-1})$
values	0.1	7	97
parameters	μ_{U_0}	μ_{U_L}	slope $(m.m^{-1})$
values	0.6	0.73	1.6 $^0/_{00}$

[1] Laboratoire de Conception et d'Intégration des Systèmes

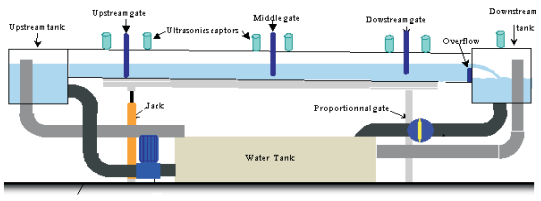


Figure 3: Pilot channel of Valence.

One experimentation has been chosen to illustrate this approach.
Note that water flow is deduced from the gate equations, and has not been measured directly. The data pictured below have been filtered to get a better idea of the experimentation results.

In each experiment, the system is initially in open loop at a steady state:

$Q_e(0) = 2.5dm^3.s^{-1}, H_e(0) = 1.1dm, H_e(L) = 1.26dm.$

The loop is closed at time $t = 50sec$ with a new set point given by:

$\bar{Q}(0) = 2dm^3.s^{-1}, \bar{H}(0) = 1.3dm, \bar{H}(L) = 1.43dm.$

Two experimentations are pictured in Fig. (4), with $\max k_0 k_L = 0.888$ and the following values:

1. $k_0 k_L = 0.247$ & $m_0 = 0$, $m_L = 0$,
2. $k_0 k_L = 0.247$ & $m_0 = -0.002$, $m_L = 0.001$.

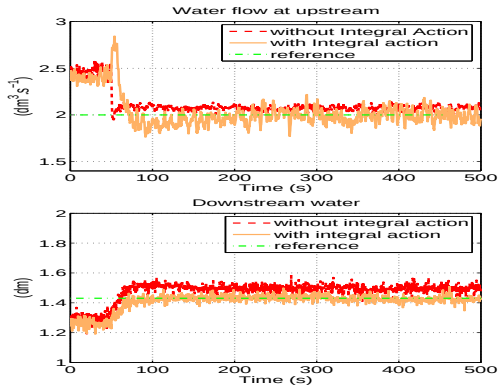


Figure 4: Water flow at upstream and level at downstream.

To conclude this part, let notice that for the micro-channel, both tests (27)-(28) are quiet equivalent (is not the case for rivers like the Sambre in Belgium). In all the cases, one conclusion is the same, the stability of the system is insured if both tests (27)-(28) and the condition (31) are realized.
Exact convergence is ensured if the integral part of the control is added even if it is applied to the real and so the non linear system.

6 CONCLUSION

In this paper, a boundary control law with integral action is proposed, as a new stability condition depending on the Riemann coordinates. Simulations and experimentations realized strengthen on the fact that the stability conditions (16)-(18) can be developed to fit to the non linear case. Improvements will be the development of the works on (Dos Santos et al., 2007) to non linear system of conservation laws, and/or to couple both previous results and generalize them to greater dimension systems.

ACKNOWLEDGEMENTS

The authors would like to thank professor E. Mendes and the LCIS to have allowed us to realize our experimentations on the micro-channel. In the same way, thanks to the Cemagref for the use of the software SIC.

REFERENCES

Coron, J. M., d'Andréa Novel, B., and Bastin, G. (2007). A strict Lyapunov function for boundary control of hyperbolic systems of conservation laws. *Automatic Control, IEEE Transactions on Automatic Control*, 52(1):2–11.

Coron, J.-M., de Halleux, J., Bastin, G., and d'Andréa Novel, B. (2002). On boundary control design for quasi-linear hyperbolic systems with entropies as Lyapunov functions. *Proceedings 41-th IEEE Conference on Decision and Control, Las Vegas, USA,*, pages 3010 – 3014.

Dos Santos, V., Bastin, G., Coron, J.-M., and d'Andréa Novel, B. (2007). Boundary control with integral action for hyperbolic systems of conservation laws: Lyapunov stability analysis and experimental validation. *submitted to Automatica*.

Dos Santos, V. and Prieur, C. (2007). Boundary control of a channel: practical and numerical studies. *in preparation*.

Litrico, X., Fromion, V., Baume, J.-P., Arranja, C., and Rijo, M. (2005). Experimental validation of a methodology to control irrigation canals based on saint-venant equations. *Control Engineering Practice*, 13:1425–1437.

Prieur, C., Winkin, J., and Bastin, G. (2006). Boundary control of non-homogeneous systems of conservation laws. *preprint*.

Renardy, M. and Rogers, R. (1993). An introduction to partial differential equations. *Springer Verlag*.

MINIMIZING THE ARM MOVEMENTS OF A MULTI-HEAD GANTRY MACHINE

Timo Knuutila, Sami Pyöttiälä and Olli S. Nevalainen
Department of Information Technology, University of Turku, FI-20014 Turku, Finland
timo.knuutila@it.utu.fi, sami.pyottiala@it.utu.fi, olli.nevalainen@it.utu.fi

Keywords: Printed circuit board, electronics assembly, multi-head placement machine, optimization, production control.

Abstract: In *printed circuit board* (PCB) manufacturing multi-head gantry machines are becoming increasingly more popular in *surface mount technology* (SMT), because they combine high speed with moderate price.
This kind of machine picks up several components from the feeder and places them on the PCB. The process is repeated until all component placements are done. In this article, a subproblem of the machine control is studied. Here, the placement order of the components, the nozzles in the placement arm and the component locations in the feeder are fixed. The goal is to find an optimal pick-up sequence when minimizing the total length of the arm movements.
An algorithm that searches the optimal pick-up sequence is proposed and tested widely. Tests show that the method can be applied to problems of practical size.

1 INTRODUCTION

The electronics industry has been one of the fastest growing fields of industry in the last 20 years. Here, one of the central fields is the manufacturing of *printed circuit boards* (PCBs) which are needed everywhere in present days. Electronic components are installed to PCBs with specialized, automated component placement machines. Production batches of similar PCBs can be very large especially in the mass production of consumer electronics. PCBs can be manufactured with a single machine or more commonly with an assembly line of several consecutive machines of different types. An assembly line consists of a solder paste printer, a few placement machines and an oven in which the components are fixed onto the board. Usually, each placement machine is specialized to place a certain set of component types.

In the past, components were attached to PCBs mainly using *through-hole technology* (THT) but nowadays, when minituarizing the products, the industry has changed over to *surface mount technology* (SMT). In order to be successful, the component placement operations require great accuracy from the placement machine, because components and PCBs have become smaller and smaller. While technology has developed, different types of placement machines have been invented, including dual-delivery, multi-station, turret-type, and multi-head gantry machines. Each machine type has its own special features and is suitable for different types of assembly tasks, see e.g. (Ayob et al., 2002) and (Ayob and Kendall, 2005) for discussion.

The operation of a placement machine is controlled with a control program which states the details of individual component placement steps. This program should force the machine perform its task as fast as possible while still satisfying high quality standards. A single PCB can include dozens of different kinds of components and the total number of components on one PCB can amount to several hundreds. The actual placement of a component requires the use of a suitable *nozzle*. One or more nozzles are attached to the placement head of a machine. Normally, nozzles can be changed when necessary. In some machines the nozzle changes have to be done manually while some others can change them automatically. The placement machine fetches components from a feeder unit capable of holding a large number of copies of components of each type and then places

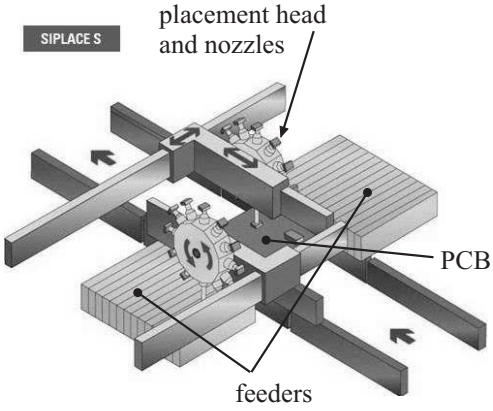


Figure 1: There are many multi-head gantry machines in Siplace-series of Siemens. (The image is taken from Siemens www-page.).

them onto the PCB being manufactured. The determination of the route the placement head travels during a placement job can be considered a special *asymmetric travelling salesman problem* (ATSP) . Solving the problem requires efficient heuristics that have been studied in (Ball and Magazine, 1988) (Leipälä and Nevalainen, 1989) for example.

In this article the optimal control of a multi-head gantry placement machine (Sun et al., 2004) is discussed. For example Siemens has this type of machines, see Figure 1 for details. Multi-head gantry machines have increased their popularity in recent times due to their flexibility and relatively low acquisition price. In this machine the PCB is positioned firmly on the stationary table. The machine has a placement arm with a placement head at one end. The placement head moves above the PCB and the component feeders in (x,y)-plane parallel to the PCB. The feeders are stationary and arranged as a linear array along the x-coordinate-side of the PCB fixation table. There are several (1-30 pieces) *spindles* (nozzle holders) in a placement head and each spindle can hold any type of nozzle. A nozzle can grab a single component at a time and different components may require certain nozzle. Every nozzle of the placement head can hold a component simultaneously. The placement head can place one component at a time onto a PCB and it must then move to the position required by the next component. While performing a placement job the machine will retrieve multiple components at a time from the feeders into the nozzles of the placement head and place them to predetermined positions on the PCB. This process will be repeated as many times as necessary. See (Pyöttiälä et al., 2005) for more details on the design of the multi-head gantry machine.

The controlling of a multi-head gantry machine can be seen as a sequence of multiple decisions. These include among others the selection of nozzles, the feeder arrangement, the component pick-up and placement sequencing. When the decisions are done consecutively, each of them depends on the previous ones. For instance the optimal component pick-up sequence depends on the beforehand selected nozzles and the order in which the components are in feeder. In this hierarchical way it is possible to find a sufficiently efficient solution for manufacturing a certain PCB but it will not guarantee a globally optimal solution in a sence of assembly time of a single job. Even though the hierarchical method is complicated it has led to some good results (Kumar and Li, 1995; Crama et al., 1990; Lee et al., 1999) whereas the joint modelling of this machine type leads to a mathematical model of impractically high complexity.

The approach used in this article is slightly different from previous studies. Our aim is not to solve the whole control problem but instead to consider a particular subproblem in greater detail. The subproblem can be chosen because it is a part of the overall manufacturing problem. All subproblems have to be solved in order to solve the entire problem. In this work, it is especially asked in which order the multi-head arm should pick up the components to be placed so that the movements of the arm are minimized for a certain printed circuit board (PCB). Two cases are discussed here. In the first case the sequence for performing the placements, the nozzles in the placement arm and the order of component reels in the feeder are predetermined and a pick-up sequence of the components and its cost shall be computed. This is called the *single pick-up event problem (SPE)*. In the second case the assumptions are as above, but the computed pick-up sequence should be the one with a minimal cost (i.e. time). This is called the *minimal pick-up sequence problem, MPS*. Here again, the optimal machine control depends on both the placement sequence and the feeder assignment at the same time, see (Leipälä and Nevalainen, 1989) for discussion of the special case with a single nozzle in the pick-up-placement head. In this paper, there are multiple nozzles in the placement head and the arm may contain multiple copies of the same nozzle type as it is presently common due to the very skewed distribution of different component types on PCBs.

The rest of the work is organized as follows. The notation and terminology are presented in section 2, as well as the actual research problems. Then in section 3 the problems are solved and possible algorithms are proposed. The best proposed algorithm is tested and the results introduced in section 4. The final section consists of the concluding remarks.

2 MINIMAL PICK-UP SEQUENCE (MPS) PROBLEM

2.1 Notation and Terminology

The sets of component types and nozzle types are denoted with $C = \{c_1, \ldots, c_n\}$ and $T = \{\tau_1, \ldots, \tau_m\}$, respectively. A *job* w is a sequence consisting of triplets (c,x,y), where $c \in C$ and the pair $x,y \in \mathbb{R}$ give the location of the component on the PCB. Let $\tau(c)$ be a function defining the *nozzle type* that has to be used to pick up a certain component of type c. Note that different component types (say c_i and c_j) may well require the use of a nozzle of the same type ($\tau(c_i) = \tau(c_j)$). The multisets of component types and nozzle types of job w are denoted with $C(w)$ and $T(w)$, respectively. They are defined as

$$\begin{aligned} C(w) &= \{c\langle i\rangle \mid (c,x,y) \in w\} \\ T(w) &= \{\tau(c)\langle j\rangle \mid c \in C(w)\}, \end{aligned}$$

where i and j give the number of copies of the particular c and $\tau(c)$, respectively. An *arm* a of *capacity* a_{max} is a sequence of nozzle types of length a_{max}. Similarly to jobs, let us denote with $T(a)$ the multiset of nozzle types in a. Given a job w and an arm a, we say that *a can pick up w* if and only if $T(w) \subseteq T(a)$, that is, there is a nozzle of a correct type in a for each component type of w. The $\subseteq$-operator is here defined between multisets (i.e. for each $\tau(c)\langle j\rangle \in T(w)$ there is $\tau(c)\langle j'\rangle \in T(a)$ such that $j \leq j'$). Note that the order in which the component types appear in w and a may be different. Consequently, this means that the placement order may differ from the pick-up order.

Example. Suppose we have a job $w = (c_1,x_1,y_1)(c_2,x_2,y_2)(c_3,x_3,y_3)(c_4,x_4,y_4)$, $\tau(c_1) = \tau(c_2) = \tau_1$, $\tau(c_3) = \tau(c_4) = \tau_2$, an arm of capacity 5, 3 nozzles of type τ_1 and 4 of type τ_2. Clearly $T(w) = \{\tau_1\langle 2\rangle, \tau_2\langle 2\rangle\}$. Arm $a_1 = \tau_1\tau_1\tau_1\tau_2\tau_2$ can pick up w, since $T(w) \subseteq T(a_1) = \{\tau_1\langle 3\rangle, \tau_2\langle 2\rangle\}$. On the other hand, arm $a_2 = \tau_1\tau_2\tau_2\tau_2\tau_2$ can not pick up w, since $T(w) \not\subseteq T(a_2) = \{\tau_1\langle 1\rangle, \tau_2\langle 4\rangle\}$.

2.2 Cost of a Pick-up Event

Suppose that we can pick up w using a. We next formalize a model for the actual execution of this process. We assume for the sake of simplicity that there is only one source for each component type in the feeder unit. Then each component type c has a unique location, say $s(c)$, ranging over the different locations in the feeder unit. Hence, s is an injective function $s\colon C \to \{1, \ldots, f_{max}\}$, where f_{max} is the total number of slots in the feeder unit.

The pick-up arm moves first to the location $s(c)$ of some component type c of the current job w. It then selects the next of the remaining components and moves to the corresponding location until all components have been picked up. The distance between two locations, say $s(c_1)$ and $s(c_2)$, is $|s(c_1) - s(c_2)|$. If we assume that the movement time between the pick-ups is linearly dependent on the distance between the pick-up locations, then it suffices to consider just these distances when defining the pick-up cost.

Let w^o be a *pick-up order* of a sequence of placement instructions w. Supposing that w can be picked up with a, we define $cost(w^o,a,s)$, the *cost of picking w up in order w^o with a given s*, as

$$cost(w^o,a,s) = MOV + \sum_{i=1}^{|w|-1} |s(w^o_{i+1}) - s(w^o_i)|,$$

where MOV represents the cost of the movement between the feeder and the board.

There are several ways to define the cost of the movement between the feeder and a PCB, see Fig. 2. At the end of every placement the arm has to return to the feeder to pick up the next set of components. After the arm has picked up these components it travels back on the PCB. We can approximate the locations of the arm on the PCB roughly using the centre of PCB only (see Fig. 2 a). The second and more gentle way to approximate the start and end positions of the pick-up phase is to use the centre point of the locations which belong to the same load of the arm (i.e. the set of components simultaneously in the placement arm) (Fig. 2 b). The third and the most accurate way is to use the exact location of the last component of previous and the first component of the next placement phase (Fig. 2 c). Note that the third method can only be used if the actual order of the component placements is known.

It is obvious that different pick-up orders w^o for the same w have in general different costs. Our aim is to select, for a given w, a and s, the ordering (permutation) w^o of w with the minimal cost. Let us denote this cost, which we call the *cost of a pick-up event for w* (given a and s), with $ecost(w,a,s)$.

Problem 1. (*Single pick-up event problem, SPE)* Given w, a and s, compute $ecost(w,a,s)$ and the associated pick-up sequence w^o for w (supposing one exists).

2.3 Cost of a Pick-up Sequence

The number of component placements per PCB is normally very large in comparison to the total number of nozzles in arm. Therefore the placements must be divided into several subjobs. Given a job

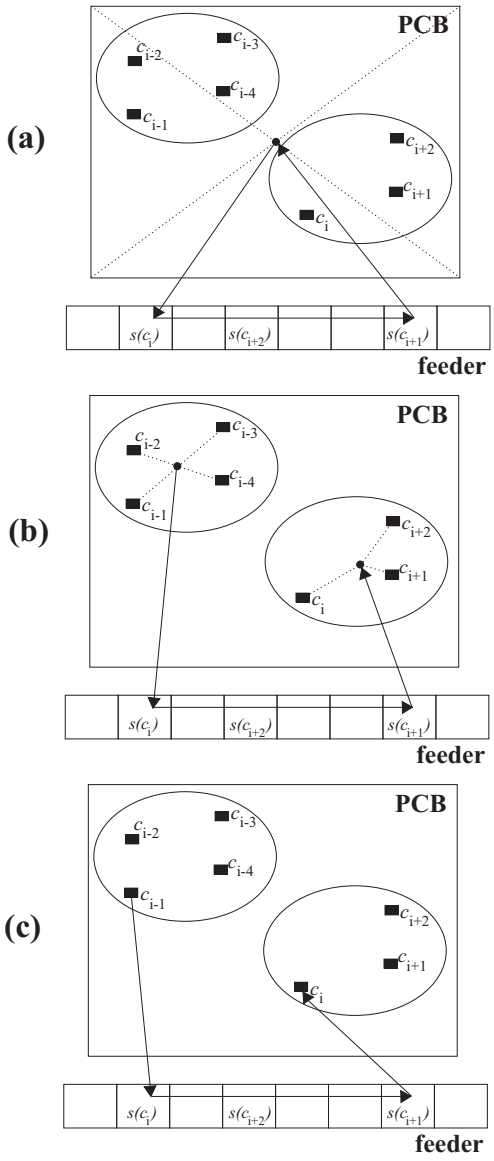


Figure 2: Three different ways to define the cost of the arm movement between the feeder and PCB.

w and an arm a, the *pick-up sequence of w using a* is any partition of w into subjobs $w_1, w_2, \ldots, w_p$ (i.e. $w = w_1 \cdot w_2 \cdots w_p$) such that a can pick up each w_i ($i = 1..p$). The *length* of such a pick-up sequence is p. The *cost of a pick-up sequence* is simply the sum of the costs $ecost(w_i, a, s)$ of individual pick-up events. One can partition w into subjobs in many different ways and the total picking cost depends on the particular partition. The *minimum cost pick-up sequence* for given w, a and s is a pick-up sequence with a minimal cost denoted by $mcost(w, a, s)$.

Problem 2. (*Minimal pick-up sequence problem, MPS)* Given w, a and s, find a pick-up sequence $w_1, w_2, \ldots, w_p$ for w such that

$$\sum_{i=1}^{p} ecost(w_i, a, s)$$

is minimal and compute the associated $mcost(w, a, s)$.

3 SOLVING THE SPE AND MPS

3.1 Pick-up Events

Let us first consider the SPE (problem 1) which determines for given w, a and s the minimal pick-up ordering w^o. This problem can be solved for small a even by a brute force -method which checks all the possible ways of feeder-to-nozzle-combinations. However, for greater arm sizes the algorithm becomes soon unpractical in special when the SPE-problem should be solved repeatedly a great number of times.

We first consider only the movements above the feeder unit and thus ignore the cost of moving to the PCB and back. It is obvious that if w contains several occurrences of the same component type, all of these should be picked up once the arm is in the appropriate location. (The is true, because the machine model does not allow so called *gang pick-ups*) Hence, the problem reduces into finding the shortest path connecting all the feeder locations

$$S(w) = \{s(c) \mid c \in C(w)\}.$$

Now let s_{min} and s_{max} be the minimal and maximal elements of $S(w)$. It is clear that the length of any path connecting all the members of $S(w)$ (points on a line segment), is at least $s_{max} - s_{min}$. Therefore, we have two obviously minimal connecting paths: the ones connecting the points of $S(w)$ in their increasing or decreasing order.

In practise, the length of the initial movement depends also on the location $s(w_1^o)$ and the initial location of the arm (the location where it was left after placing the last component of the previous placement). Similarly, the last pick-up is followed by the movement to the first placement location on the PCB. The *placement order* of w defines what these locations exactly are. If the placement orders of all pick-up events $w_1, w_2, \ldots, w_p$ are known beforehand, the initial location of the arm, when event w_i is started, is the last placement of the previous event w_{i-1}, and the location we move after the pick-up of w_i can be found from the first placement of w_i.

However, the placement order is not necessarily known to us. Formerly the reason for this was that

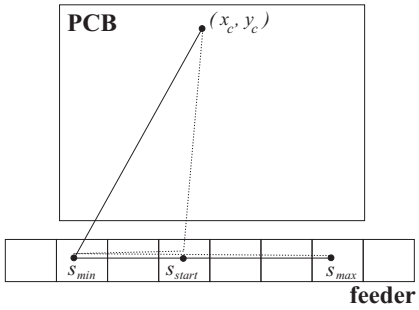


Figure 3: The linear ordering of the pick-up points gives us the minimal path.

the pick-up and the placement sequencing problems were interconnected so that both of them should be solved jointly. Currently, another reason for the lack of this information has arisen: There are placement machines which decide the printing order automatically at the pick-up phase. The best we can then do is to approximate the initial and final locations with the center of the PCB or with the average of the placement locations. In the former case, we use the same constant location in all cost calculations, in the latter case the center point is specific to each pick-up event. Let us denote this center point with (x_c, y_c).

Consider the three pick-up points s_{min}, s_{max} and s_{start}, where s_{start} is the first pick-up place. If $s_{start} \notin \{s_{min}, s_{max}\}$, suppose (without loss of generality) that

$$s_{start} - s_{min} \leq s_{max} - s_{start},$$

i.e. the start position is closer to the smallest position than to the largest one. The minimal route on top of the feeder goes now first from s_{start} to s_{min} (picking all the components on the way) and then all the way to s_{max}. The difference to the length of a route starting directly at s_{min} is hence $s_{start} - s_{min}$. Although the distance from (x_c, y_c) to $(s_{start}, 0)$ might be shorter than to $(s_{min}, 0)$, the difference is always smaller than $s_{start} - s_{min}$ due to the triangular inequality (consider the triangle with corners at (x_c, y_c), $(s_{min}, 0)$, and $(s_{start}, 0)$), see Fig. 3. Hence, the linear ordering of the points gives us the minimal path even in this case. If the arm is moved by two motors, one for the *x*-directional and one for the *y*-directional movement and these operate at the same speeds, the movement time is related to the maximum of the coordinate distances (the Chebyshev distance). Naturally, the triangular inequality holds here, too.

The SPE can now be solved trivially by sorting the locations $s(w_i)$ into ascending order. The smallest and largest positions are then $s(w^o_1)$ and $s(w^o_{|w|})$, respectively. The value of $ecost(w, a, s)$ is $d((x_c, y_c), (s(w^o_1), 0)) + s(w^o_{|w|}) - s(w^o_1) + d((s(w^o_{|w|}), 0), (x_c, y_c))$, where d is an appropriate metric. If the movements between the PCB and the feeders are ignored (or the cost is constant) the formula simplifies to $s(w^o_{|w|}) - s(w^o_1)$.

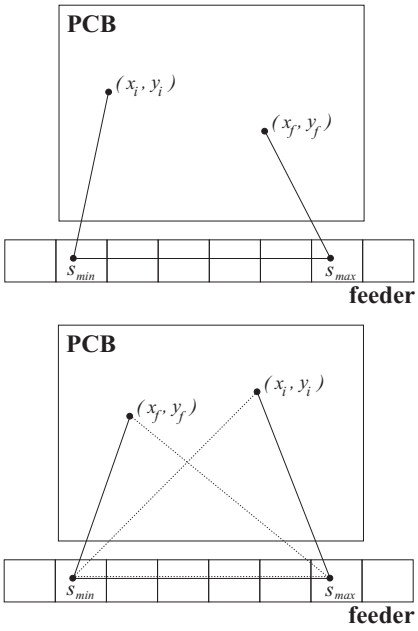


Figure 4: The arrangement of components depends on x_i and x_f.

Suppose now that the printing order is known, and that (x_i, y_i) and (x_f, y_f) are the initial and final location of the arm in some pick-up event. That is, (x_i, y_i) is the location where the last component of the previous pick-up event was placed and (x_f, y_f) is the location where the first component of this pick-up event will be placed. A similar geometrical analysis gives again (see Fig. 4), that

- if $x_i \leq x_f$, the component pick-ups should be arranged in an ascending order of $s(c)$ and
- if $x_i > x_f$, the component pick-ups should be arranged in a descending order of $s(c)$.

The corresponding value of $ecost(w, a, s)$ is then $d((x_i, y_i), (s(w^o_1)), 0)) + |s(w^o_{|w|}) - s(w^o_1))| + d((s(w^o_{|w|}), 0), (x_f, y_f))$.

3.2 Pick-up Sequences

The task in the MPS is to partition a large w into subjobs $w_1, w_2, \ldots, w_p$ such that the accumulated pick-up cost is minimal. Note first that the greedy partitioning that gives minimal number of pick-ups (Knuutila et al., 2007) (pick up always as many components as possible) does not always give an minimal solution

when the goal is to minimize the length of arm movements. However, the greedy algorithm still minimizes the number of pick-up rounds. Consider the following example:

$$w = (c_1, x_1, y_1), \ldots, (c_5, x_5, y_5),$$
$$s(c_1) = 1, s(c_2) = 2, s(c_3) = 100,$$
$$s(c_4) = 101, s(c_5) = 102,$$
$$\tau(c_i) = \tau \text{ for } i = 1..5, \text{ and}$$
$$a_{max} = 3.$$

The greedy partitioning would first pick component types c_1, c_2, c_3 and then c_4, c_5. The simplest definition of $ecost(w,a,s)$ would give a cost of $(100-1)+(102-101) = 100$, whereas the cost for partition c_1, c_2 and c_3, c_4, c_5 would be $(2-1)+(102-100) = 3$.

The same counter example gives the following costs for the second definition of $ecost(w,a,s)$. In the greedy case

$$d((x_c,y_c),(1,0)) + (100-1) + d((x_c,y_c),(100,0)) + d((x_c,y_c),(101,0)) + (102-101) + d((x_c,y_c),(102,0)),$$

and in the alternative case

$$d((x_c,y_c),(1,0)) + (2-1) + d((x_c,y_c),(2,0)) + d((x_c,y_c),(100,0)) + (102-100) + d((x_c,y_c),(102,0)).$$

Considering the geometry of the feeder (a straight line segment), distances to adjacent slots from the PCB center are practically the same. Hence, the greedy distance is approximately

$$d((x_c,y_c),(1,0)) + 3 * d((x_c,y_c),(100,0)) + 100,$$

and the alternative is

$$2 * d((x_c,y_c),(1,0)) + 2 * d((x_c,y_c),(100,0)) + 3.$$

If the distances to feeder slot 1 and slot 100 are approximately the same from the PCB center (e.g. they are at the left and right ends of the feeder), then the alternative partitioning is clearly a winner here, too. A similar inspection can be carried out also for the third definition of $ecost(w,a,s)$.

3.2.1 Brute Force Method

Solving the MPS seems to lead to an exhaustive search over all possible ways of partitioning w into subsequences. Procedure `sequence` implements a brute-force algorithm for this problem. The global variables `best` and `bestcycle` store the value and the partition of the best solution found so far. The current partition is given by the array `cycle`. Argument `p` gives the level of a current recursive call and `cost` expresses the *ecost* of the subjobs 1 to (p-1). Algorithm `sequence` is initially called as `sequence(1,0)` with `cycle[0] = 1` since at the first pick-up cycle at least one component has to be picked up and the cost should be zero before any components has been picked up. At the beginning, `best` is initialized to some large value. After the execution, the result is in array `bestcycle`.

```
// cycle = array of the start indexes of
// pick-up cycles
// p = current pick-up cycle
// N = length of the placement sequence w

sequence (int p, int cost)
    int i, k, x;

    k := cycle[p - 1]; // start of the
                       // previous cycle

    // i = tentative start of the next cycle
    FOR i = k+1 TO k + pick-up head size DO
        IF the pick-up head can pick components
           from k to i - 1 THEN
            x := ecost(k, i - 1);
            IF ((cost + x) < best) THEN
                IF i = N+1 THEN // solution
                    best := cost + x;
                    bestcycle := cycle;
                ELSEIF i <= N
                    cycle[p] = i;
                    sequence(p + 1, cost + x);
    END END END END
END
```

The greedy method to form a pick-up sequence (introduced in (Knuutila et al., 2007)) minimizes the number of pick-up-cycles for given w and a. The minimal solution (in terms of total length of arm movements) found by `sequence` seems always to be clearly better than that generated by the greedy method when measuring the performance using the length of tour the placing arm has to travel to pick up all components for a certain job. A problem here is that `sequence` is capable of solving only small problems in which the job size is few dozens of components; its running time explodes for placement tasks of practical size.

3.2.2 Dynamic Programming

We apply dynamic programming to search the minimal solution in a fast way. Fast algorithm is required also for this subproblem since searhing for a minimal pick-up sequence will be an important part of higher level optimization software. Consider job w of length n. Dynamic programming can be applied since the

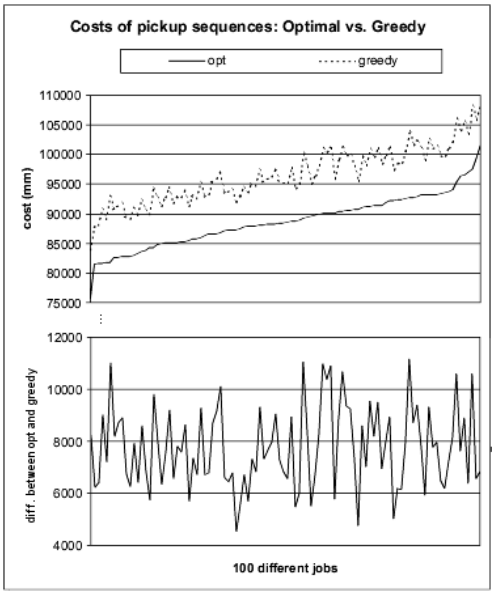


Figure 5: The pick-up sequence generated by the greedy method is clearly worse than the minimal solution found by exhaustive search when measuring the length of arm movements in mm. We tested here 100 different jobs of length 600 with arm size of 20, 7 seven different nozzles, and 20 (also the number of feeders) different components per job.

solution for any suffix w_k of job w must be minimal regardless the solution of the earlier subjobs w_i $(i < k)$ which must be minimal, too. The minimal solutions for all starting points of subjobs are searched and memorized starting from the last one (the subjob of length 1) and proceeding towards longer subjobs. Finally, the minimal solution for the subjob of length n (the actual job) is found. For this method the running time $T(n, a_{max})$ is $O(na_{max}^2)$ in the worst case and the memory usage $M(n) = \theta(n^2)$.

Procedure `dynamic_sequence` uses global arrays `comp_costs` and `comp_parts`; `comp_parts` stores the minimal cycle sequences for every suffix of a job and `comp_costs` holds the costs of these sequences. The procedure uses backward recursion by starting from the end of the job and proceeding towards the beginning. At each step `j`, `round_min` is set to some big value and the maximum number of components that the placing arm can hold simultaneously starting from the `j`'th placement instruction is stored into `a_can`. Then, *ecost*s for all possible pick-ups of components from `j` to `(j + a_can)` are calculated in turn and summed with the costs of corresponding suffixes stored in `comp_costs`. List `part_list` and variable `round_min` keep the information of the best cycle sequence of the current round. `Part_list` is formed by linking the minimal cycle sequence list of a corresponding suffix on the right of the current one. Finally, the `bestcycle` (as in the previous algorithm) is constructed on the basis of `comp_parts[1]` and the minimal cost of the pick-up sequence for the job is returned.

```
// bestcycle = array of the start indexes of
//             the pick-up cycles for the
//             minimal cycle sequence
// N = length of the placement sequence w
// part_list = linked list, integers as items
// comp_costs = array of minimum cost for all
//              possible job suffixes
//              starting positions
//              between [1, N]
// comp_parts = array of length N of linked
//              lists which items determine
//              the minimums stored
//              into comp_costs

dynamic_sequence : int
    int i, j, x, round_min;

    FOR j := N DOWNTO 1 DO
        round_min := positive infinity;

        a_can := max number of components
                 that arm can hold at once
                 starting from j'th placement
                 instruction;

        FOR i := 1 TO a_can DO
            x := ecost(j, j + i - 1);
            IF j + i - 1 < N THEN
                x := x + comp_costs[j + i];
            END
            IF x < round_min THEN
                round_min := x;
                create new part_list;
                part_list.put_right(i);
                IF j + i - 1 < N THEN
                    part_list.append(
                      comp_parts[j + i]);
                END
            END
        END
        comp_costs[j] := round_min;
        comp_parts[j] := part_list;
    END
    construct bestcycle -table
      by comp_parts[1] -list;
    RETURN comp_costs[1];
END
```

The suffix of a job is independent of the prefix of the job only in cases (a) and (c) of Fig. 2. Therefore, the dynamic programming approach cannot be applied if the placement locations are approximated as in case (b). However, case (c) is sufficient; in practice the component locations are usually known.

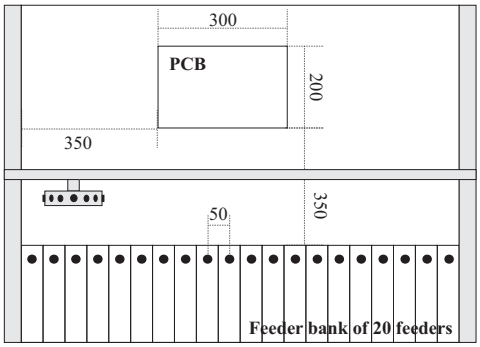


Figure 6: Some essential dimensions of a typical multi-head gantry in mm. The distances are realistic and they represent existing gantry-machines.

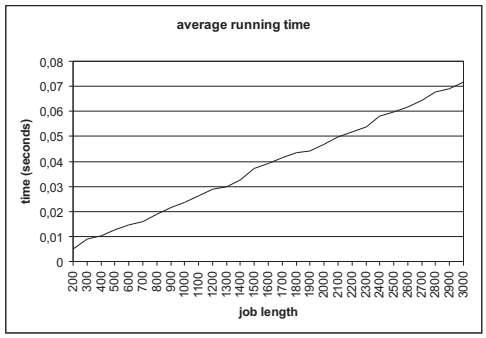


Figure 7: The average running times of procedure `dynamic_sequence.` for jobs of different length. The results are for each job length averages of $N = 100$ test jobs generated by the Markov-model described earlier.

4 RESULTS OF EXPERIMENTAL TESTS

In this section it is demonstrated that algorithm `dynamic_sequence` is fast enough to solve problems appearing in practice. It is also experimented how the cost of a pick-up sequence behaves when the arm size varies. For evaluating procedure `dynamic_sequence` we use a *Markov-model* with 20 states to generate challenging component placement tasks, see (Pyöttiälä et al., 2005) and (Pyöttiälä et al., 2006). The model is fully connected and the transition probabilities vary in wide range. As a result of this data generation model, the component type of the *i*th placement depends on the types of previous components and the number of different component types is non-uniform. However, the (x,y)-pairs (component positions on a PCB) of placement instructions are still uniformly distributed over the PCB in our test data generator.

The dimensions of a the placement machine are indicated in Fig. 6. Independent step motors move the placement head in *x*- and *y*-directions with same speed.

In all tests, the nozzles of a placing arm are selected using the uniform distribution -based heuristic introduced in (Pyöttiälä et al., 2006). In this heuristic, different types of nozzles are chosen into the arm in the same ratio as the nozzle type requirements occur in the placement job. The running times are measured in real-time seconds.

The average running times of the method of section 3.2.2 for 100 jobs of each different length (job length classes were 200, 300, 400,...,3000) are shown in Fig. 7. The tests were performed for 20 different component types, 7 nozzle types and arm size of 14.

The average running times show a clear linear tendency on the number of placements in the job. Figure

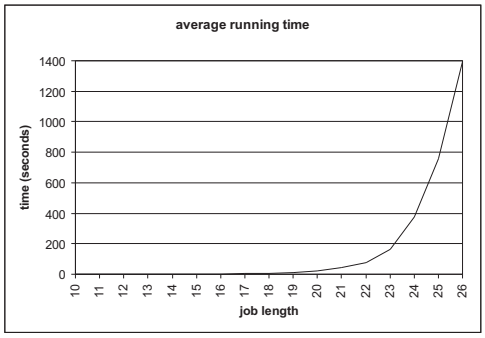


Figure 8: The running times of *brute force.* -method for jobs of different length.

9 shows how the running time of the method based on dynamic programing changes when the arm capacity is increased from 7 to 30. Again the running times are averages of 100 different jobs for each arm size. Job length was 600, the number of different components 20, and the number of different nozzle types 7. Note, that in both figures (7 and 9) the shape of the curve correlates well to the complexity class of dynamic program. However, figure 8 shows that *brute force*-based method cannot be used in practice.

Figure 10 demonstrates the decrease of the number of pick-ups when the arm size increases. Here the numbers of pick-ups are the ones of minimal pick-up sequences. The minimal cost of pick-up sequence decreases notably when the number of nozzles in a placing arm increases. Figure 11 shows this effect for job length 600, component types 20, different nozzles 7, and 100 different jobs for every arm size between 7-30).

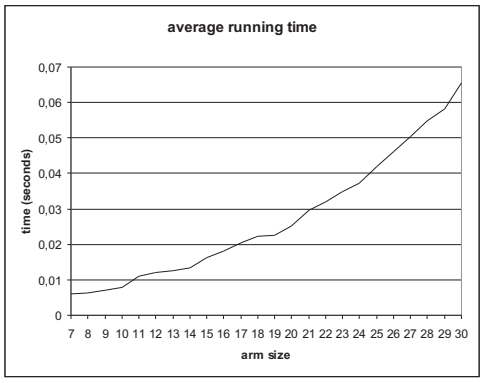


Figure 9: The average running times of varying arm sizes. Averages are for $N = 100$ randomly generated jobs for each arm sizes.

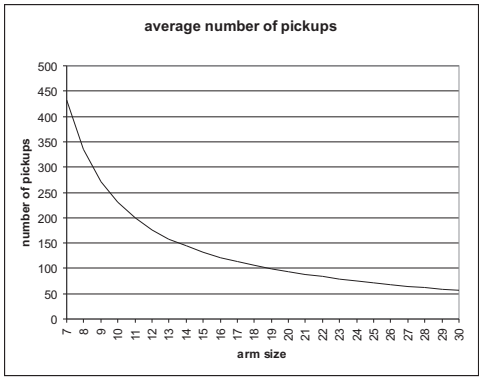


Figure 10: The average number of pick-ups for varying arm sizes.

5 CONCLUSION

In this research the subproblem relating to the control of a multi-head gantry placing machine was considered. Unlike the usual approach to this topic, this article focused on a situation where the placement order of the components is fixed and the pick-up order of the components was to be determined.

Multiple nozzles in the placement arm, the linear ordering of the different components in the feeder unit, and the fact that different types of components require different nozzles make this problem complex. In this case, the greedy algorithm that minimizes the number of component pick-ups does not, opposite to expectations, yield the minimal length pick-up-route of the placement arm. An efficient algorithm that applies dynamic programming was developed for searching the minimal pick-up sequence. The running time of the algorithm is linear on the number of components to be placed and quadratic on the arm size.

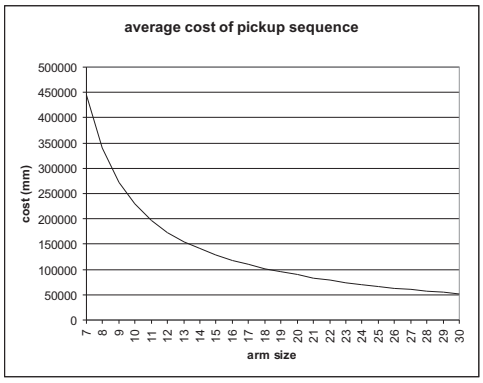


Figure 11: The average cost of pick-up sequences for varying arm sizes.

REFERENCES

Ayob, M., Cowling, P., and Kendall, G. (2002). Optimisation for surface mount placement machines. In *University of Nottingham.*

Ayob, M. and Kendall, G. (2005). A survey of surface mount device placement machine optimisation: Machine classification. In *Computer Science Technical Report No. NOTTCS-TR-2005-8*. University of Nottingham.

Ball, M. and Magazine, M. (1988). Sequencing of insertions in printed circuit board assembly. In *Operations Research 36(2), pp. 192-201*.

Crama, Y., Kolen, A., Oerlemans, A., and Spieksma, F. (1990). Througput rate optimization in the automated assembly of printed circuit boards. In *Annals of OR, Vol. 26, pp.455-480*.

Knuutila, T., Pyöttiälä, S., and Nevalainen, O.S. (2007). Minimizing the number of pickups on a multi-head placement machine. In *The Journal of the Operational Research Society.*

Kumar, R. and Li, H. (1995). Integer programming approach to printed circuit board assembly time optimization. In *IEEE Trans. Components, Packaging and Manufacturing Technology, part B: Advanced Packaging, Vol. 18 pp. 720-727*.

Lee, S., Lee, H., and Park, T. (1999). A hierarchical method to improve the productivity of a multi-head surface mounting machine. In *Proc. of the IEEE International conference on Robotics and Automation, Detroit, Michigan.*

Leipälä, T. and Nevalainen, O. (1989). Optimization of the movements of a component placement machine. In *European Journal of Operational Research,38, pp. 167-177*.

Pyöttiälä, S., Knuutila, T., Johnsson, M., and Nevalainen, O.S. (2005). Improving the pickups of components on a gantry-type placement machine. In *TUCS Technical Report 692*. Turku Centre for Computer Science.

Pyöttiälä, S., Knuutila, T., and Nevalainen, O.S. (2006). The selection of nozzles for minimizing the number of pick-ups on a multi-head placement machine. In *GTCM2006 Conference, Groningen, The Netherlands*.

Sun, D., Lee, T., and Kim, K. (2004). Component allocation and feeder arrangement for a dual-gantry multi-head surface mounting placement tool. In *International Journal of Production Economics 95 (2005) 245-264*.

POSTERS

BEHAVIOR NAVIGATION LEARNING USING FACL ALGORITHM

Abdelkarim Souissi and Hacene Rezine
EMP , Bordj Elbahri, Alger, Algerie
aks752005@yahoo.fr, rezine_hacene_emp@yahoo.fr

Keywords: Mobile Robot Navigation, Reactive Navigation, Fuzzy Control, Reinforcement learning, Fuzzy Actor Critic Learning.

Abstract: In this article, we are interested in the reactive behaviours navigation training of a mobile robot in an unknown environment. The method we will suggest ensures navigation in unknown environments with presence off different obstacles shape and consists in bringing the robot in a goal position, avoiding obstacles and releasing it from the tight corners and deadlock obstacles shape. In this framework, we use the reinforcement learning algorithm called Fuzzy Actor-Critic learning, based on temporal difference prediction method. The application was tested in our experimental PIONEER II platform.

1 INTRODUCTION

In this article, we propose a reinforcement training method where the apprentice explores actively its environment. It applies various actions in order to discover the states causing the emission of rewards and punishments. The agent must find the action which it must carry out when it is in a given situation It must learn how to choose the optimal actions to achieve the fixed goal. The environment can punish or reward the system according to the applied actions. Each time that an agent applies an action, a critic gives him a reward or a penalty to indicate if the resulting state is desirable or not (Sutton, 1998), (Glorennec, 2000). The task of the agent is to learn using these rewards the continuation of actions which gets the greatest cumulative reward.

Mobile robotics is a privileged application field of the training by reinforcement (Fujii, 1998), (Smart, 2002), (Babvey, 2003). This established fact is related to the growing place which takes, since a few years, an autonomous robotics without knowledge of the environment. The goal is then to regard behaviour as a function of mapping sensor-effector. The training in robotics consists of the automatic modification of the behaviour of the robot to improve its behaviour in its environment. The behaviours are synthesized starting from the simple definition of objectives through a reinforcement function.

The considered approach of the robot navigation using fuzzy inference as apprentice is ready to integrate certain errors in the information about the system. For example, with fuzzy logic we can process vague data. The perception of the environment by ultrasounds sensors and the reinforcement training thus prove to be particularly well adapted one to the other (Beom, 1995), (Fukuda 1995), (Jouffe, 1997), (Faria, 2000).

It is very difficult to determinate correct conclusions manually in a large base rule FIS to ensure the releasing from tight corner and deadlock obstacles, even when we use a gradient descent method or a potential-field technique due to the local-minimum problem. In such situations the robot will be blocked.

Behaviours made up of a fusion of a «goal seeking» and of an "obstacle avoidance» issues are presented. The method we will suggest ensures navigation in unknown environments with presence off different obstacles shape, the behaviour will be realised with SIF whose conclusions are determined by reinforcement training methods. The algorithms are written using the Matlab software after having integrated, in a Simulink block, the functions of perception, localization and motricity of the robot.

The application was tested in our experimental platform PIONEER II.

2 FACL ALGORITHM

We have selected a zero-order Takagi-Sugeno FIS apprentice due to its simplicity, universal approximator characteristics, generalization capacity and its real time applications. The input variables have a triangular and trapezoidal membership functions.

The SIF thus consists of *N* rules of the following form (Glorennec, 2000(:

If situation then Y1 = v[i]	and Y2= u[*i*, 1] with q[*i*,1]
	Y2= u[*i*, 2] with q[*i*,2]
	Y2= u[*i*, *j*] with q[*i*,*j*]
Critic	actor

In FACL algorithm (Jouffe, 1997(, each rule R_i of the apprentice has:

- a conclusion v_i used for the approximation of the evaluation function V^{π} of the current policy. "initialized to zeros".
- a set of discrete actions U_i identical for all rules.
- a vector of parameters q^i indicating the quality of the various discrete actions available and intervening in the current policy definition.

The characteristics of the input variables membership functions (number, position) are fixed. The number of rules is also fixed. Thus, the only modifiable characteristics of the apprentice are the conclusions v_i (critic) and the election of an action u[*i*,*j*] among *J* actions available (actor) in the rule R_i.

The FACL algorithm uses two types of training: temporal differences for training the critic reinforcement's predictions, and a competition process between available actions for the actor (Jouffe, 1997).

2.1 Critic

The role of the critic is to approximate the evaluation function which constitutes a better criterion for the choice of the actions than that represented by the primary reinforcements. The critic value in the state S_t is inferred from the conclusions vector v_t

$$V_t(S_t) = \sum_{R_i \in A_t} v_t^i . \alpha_{R_i}(S_t) = v_t . \phi_t^T \quad (1)$$

Where ϕ_t^T represents the apprentice perception at time step *t* (i.e. contains the truth value of activated rules A_t for the state S_t).

The approximation error of the critic is given by the temporal difference error TD as follows:

$$\tilde{\varepsilon}_{t+1} = r_{t+1} + \gamma . V_t(S_{t+1}) - V_t(S_t), \quad (2)$$

The critic uses this error to update the conclusions vector v by a traditional stochastic gradient descent:

$$\begin{aligned} v_{t+1} &= v_t + \beta . \tilde{\varepsilon}_{t+1} . \nabla_v V_t(S_t), \\ &= v_t + \beta . \tilde{\varepsilon}_{t+1} . \phi_t \end{aligned} \quad (3)$$

2.2 Actor

Concerning the actor, the local actions are elected for each rule activated on the basis of quality of these actions, and also according to a policy exploration which we will see further.

The global action for the state S_t is then determined by the inference of these locally elected actions:

$$\begin{aligned} U_t(S_t) &= \sum_{R_i \in A_t} Election_{U_i}(q_t^i) . \alpha_{R_i}(S_t) \\ &= Election(q_t) . \phi_t^T \end{aligned} \quad (4)$$

Where ***Election*** is a function returning the action elected for each activated rule.

The training of the optimal policy thus consists in adjusting the vector of parameters *q* so that the induced policy is improved.

Again the TD error provides a measurement of this quality improvement. Then, we obtains the following rule for training the actor

$$q_{t+1}^i(U_t^i) = q_t^i(U_t^i) + \tilde{\varepsilon}_{t+1} . \alpha_{R_i}(S_t), \forall R_i \in A_t \quad (5)$$

The above expression shows that a positive error TD implies that the action has been just applied is preferable than the t-optimal action. It is necessary to increase the quality of the action being applied.

Reciprocally, if the TD error is negative, it is necessary to decrease the quality of the action being applied because it led the system in a state whose evaluation is lower than that expected.

2.3 Eligibility Traces

The traces implementation of the critic and the actor rise directly from the incremental version of the TD.

Let $\overline{\phi}_t$ be the trace of the critic at step *t*, $\overline{\phi}_t$ is a short term memory of the visited states by the apprentice. This memory is based on the apprentice perception, i.e.: truth values of the rules:

$$\begin{aligned}\overline{\phi}_t &= \sum_{k=0}^{t} (\gamma\lambda)^{t-k}.\phi_k, \\ &= \phi_t + \gamma\lambda \sum_{k=0}^{t-1} (\gamma\lambda)^{t-1-k}.\phi_k, \\ &= \phi_t + \gamma\lambda.\overline{\phi}_{k-1,} \end{aligned} \quad (6)$$

where λ is the proximity factor of the critic.

An equivalent trace for the actor consists in memorizing the actions applied in the states. We use a short term memory of the truth values of each rule according to each action available in these rules. Let $e_t^i(U^i)$ be the trace value of the action U^i in the rule R_i at the step *t* (Jouffe, 1997):

$$e_t^i(U^i) = \begin{cases} \gamma\lambda'.e_{t-1}^i(U^i) + \phi_t^i, (U^i = U_t^i), \\ \gamma\lambda'.e_{t-1}^i(U^i), \quad else \end{cases} \quad (7)$$

where λ' is the proximity factor of the actor.

The updates of the parameters preset by (3) and (5) for all the rules and actions become:

$$v_{t+1} = v_t + \beta\tilde{\varepsilon}_{t+1}\overline{\phi}_t, \quad (8)$$

$$q_{t+1} = q_t + \tilde{\varepsilon}_{t+1}e_t, \quad (9)$$

2.4 Description of the FACL Execution Procedure

The execution in a time step can be divided into six principal stages. Let *t+1* the step of the current time; the apprentice then has applied the action U_t elected in the previous time step and on the other hand has received the primary reinforcement r_{t+1} for the transition from the state S_t to S_{t+1}. After the calculation of the truth values of the rules, the six stages are as follows (Jouffe, 1997):

1- Calculation of the t-optimal evaluation functions of the current state by the critic:

$$V_t(S_{t+1}) = v_t.\phi_{t+1}^T \quad (10)$$

As proposed by Baird (Baird, 1995), to eliminate the instability source due to the approximation by SIF, we do not perform a gradient descent on the residual of Bellman defined by ε but on the *average residual quadratic of Bellman.* The modification to be made on the trace of eligibility is then given by the following relation:

$$\overline{\phi}_t \leftarrow \overline{\phi}_t - \gamma\rho\phi_{t+1}, \ \rho \in [0,1] \quad (11)$$

2- TD Error calculation:

$$\tilde{\varepsilon}_{t+1} = r_{t+1} + \gamma.V_t(S_{t+1}) - V_t(S_t), \quad (12)$$

3- Update of training rates corresponding to the critic parameters for all rules in three stages:

In order to accelerate its training speed and to avoid instability, we adopt a heuristic adaptive training rate (Jouffe, 1997). The implementation of heuristic is carried out by the means of Delta Bar Delta rule:

$$\begin{aligned} &- \quad \delta_t^i = \tilde{\varepsilon}_{t+1}.\overline{\phi}_t^i \\ &- \quad \beta_{t+1}^i = \begin{cases} \beta_t^i + k & if \quad \overline{\delta}_{t-1}^i.\delta_t^i > 0, \\ \beta_t^i(1-\psi) & if \quad \overline{\delta}_{t-1}^i.\delta_t^i < 0, \\ \beta_t^i & else \end{cases} \\ &- \quad \overline{\delta}_t^i = (1-\psi)\delta_t^i + \psi\overline{\delta}_{t-1}^i \end{aligned} \quad (13)$$

Where in our case:

- β_t^i is the training rate of the critic for the rule R_i in the time step t;

- $\delta_t^i = \tilde{\varepsilon}_{t+1}.\overline{\phi}_t^i$ represent the variation brought to the parameter of the critic in which we propose to integrate the trace of eligibility directly and not simply the partial derivative;

- $\overline{\delta}_t^i = (1-\psi)\delta_t^i + \psi\overline{\delta}_{t-1}^i$ represent the exponential average

This rule increases linearly the training rates in order to prevent that it do not become too large, and decrementing it in an exponential way to ensure that it decrease quickly.

4- Training of the critic and the actor by updating the vector v and the matrix q :

$$v_{t+1} = v_t + \tilde{\varepsilon}_{t+1}.\beta_{t+1}.\bar{\phi}_t^T \quad (14)$$

$$q_{t+1} = q_t + \tilde{\varepsilon}_{t+1} e_t \quad (15)$$

5- It is again necessary to calculate the t-optimal evaluation function of the current state by the critic but this time with the lately updated parameters:

$$V_{t+1}(S_{t+1}) = v_{t+1}.\phi_{t+1}^T \quad (16)$$

This value will be used for the TD error calculation in the next step of time.

6- Now it remains the choice of the action to be applied in the state S_{t+1} .

We consider the case of continuous type actions, which is inferred from the various actions elected in each rule,

$$U_{t+1}(S_{t+1}) = \sum_{R_i \in A_{t+1}} Election_U(q_{t+1}^i).\alpha_{R_i}(S_{t+1}), \forall U \in U,$$

Where Election is defined by:

$$Election_U(q_{t+1}^i) = ArgMax_{U \in U}(q_{t+1}^i(U) + \eta^i(U) + \rho^i(U)) \quad (17)$$

The traces eligibility updates for the critic and the actor are given by the two following formulas:

$$\bar{\phi}_{t+1} = \phi_t + \gamma\lambda.\bar{\phi}_t, \quad (18)$$

$$e_{t+1}^i(U^i) = \begin{cases} \gamma\lambda'.e_t^i(U^i) + \phi_{t+1}^i, (U^i = U_{t+1}^i), \\ \gamma\lambda'.e_t^i(U^i), \quad else \end{cases} \quad (19)$$

2.5 Proposed Exploration / Exploitation

The actions election strategy which we used is a combination of directed and random exploration (Jouffe, 1997). The global election function, applied to each rules, is then defined by:

$$Election_U(q) = ArgMax_{U \in U}(q(U) + \eta(U) + \rho(U)) \quad (20)$$

Where U represents the set of available discrete actions in each rules, and q the associated vector quality. $\eta(U)$ the random exploration term, and $\rho(U)$ the directed exploration term.

The term η is in fact a random values vector. It corresponds to a vector ψ of values sampled according to an exponential law, standardized in order to take into account the q qualities size scale:

$$s_f(U) = \begin{cases} 1 \quad si \quad Max(q(U)) = Min(q(U)) \\ \dfrac{s_p(Max(q(U)) - Min(q(U)))}{Max(\psi)}, \quad else \end{cases} \quad (21)$$

$$\eta(U) = s_f(U).\psi, \quad (22)$$

Where s_p represents the maximum size of the noise relative to the qualities amplitude, s_f is the corresponding normalisation factor. Used alone, this term of exploration allows a random choice of actions when all qualities are identical and allows in the other case, to test only the actions of which quality satisfied:

$$q(U) \geq q(U^*) - s_p(Max(q(U)) - Min(q(U))) \quad (23)$$

This term then allow us to avoid the choice of bad actions (Jouffe, 1997).

The term of directed exploration ρ permit the test of actions not having been applied often. It is thus necessary to memorize the times number where the action was elected. This term is defined in the following way: $\rho(U) = \dfrac{\theta}{e^{n_t(U)}}$ (24)

Where θ represents a positive factor which permit to equilibrate the directed exploration, $n_t(U)$ the applications number of action U at the time step *t*. In the case of actions of the continuous type, $n_t(U)$ corresponds to a number of applications of the discrete action U in the considered rule. Let U_t^i the discrete action elected at the step of time t in the rule R_i ; the update of this variable is determined by the following equation:

$$n_t(U^i) = n_{t-1}(U^i) + 1, \quad \forall R_i \in A_t, U^i = U_t^i \quad (25)$$

3 EXPERIMENTAL PLATFORM

Pioneer P2-dx with its modest size lends itself well to navigation in the tight corners and encumbered

spaces such as the laboratories and small offices. It has two driving wheels and a castor wheel.

Figure 1: Pioneer II Robot Photo.

To detect obstacles, the robot is equipped with a set of ultrasounds sensors. It supports eight ultrasonic sensors placed in its front (fig. 2). The range of measurements of these sensors lies between 10cm as minimal range and 5m as maximum range.

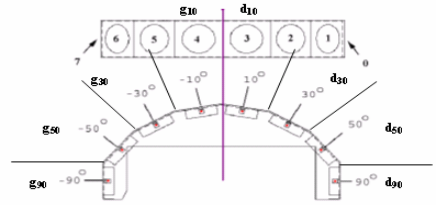


Figure 2: Position of the ultrasonic sensors used in the robot Pioneer II.

The software Saphira/Aria (Konolige, 2002a), (Konolige, 2002b) allows the control of the robot (C/C++ programming). We have integrated functions of Saphira (perception, localization and motricity) in Simulink using API (S-Function), Thus we can benefit from MatLab computing power and simplicity of Simulink to test our algorithms and to control the robot Pioneer II.

4 EXPERIMENTATION

The task of the robot consists in starting from a starting point achieving a fixed goal while avoiding the static obstacles of convex or concave type. It is realised by the fusion of two elementary behaviours «*goal seeking*» and «*obstacle avoidance* ".

4.1 The «goal seeking» Behaviour

For the "goal seeking" behaviour, we consider three membership functions for the input θ_{Rb} (robot-goal angle), and two membership functions for the input ρ_b (robot-goal distance) (Figure 3). The base of knowledge consists of six fuzzy rules. The SIF controller has two output for the actor (rotation speed Vrot and the translation speed Vit) and one output for the critique who is related to the evaluation function of the pair actions (Vrot,Vit).

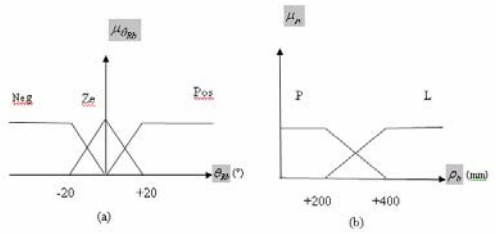


Figure 3: membership functions for θ_{Rb} (a) and ρ_b (b).

From the heuristic nature of FACL algorithm, we carried out several tests to determine the values of the parameters which accelerate the training speed and to obtain good performances for the apprentice. After a series of experiments, we found the following values:

$\theta = 50, \lambda = \lambda' = 0.9, \rho = 0.5$, $sp = 0.1$ and $\gamma = 0.9$

Available actions for all rules, are as follow {-20°/s, -10°/s, -5°/s, 0°/s, +5°/s, +10°/s, +20°/s} for the rotation velocity and {0 mm/s, 150 mm/s, 350mm/s} for translation, which gives 21 possible actions in each fuzzy rule.

The reinforcement function is defined as follows:

- If the robot is far from the goal, the reinforcement is equal to

- 1 if ($\theta_{Rb} . \dot{\theta}_{Rb} < 0$) & Vit=0
- 1 if (-1°< θ_{Rb} <+1°) & Vit $\neq 0$
- 0 if ($\theta_{Rb} . \dot{\theta}_{Rb} = 0$) & Vit=0
- -1 else

- If the robot is close to the goal, the reinforcement is equal to

- 1 if ($\theta_{Rb} . \dot{\theta}_{Rb} < 0$) & Vit=0
- -1 else

Figure (4) shows the trajectory of the robot during the phase of training and validation.

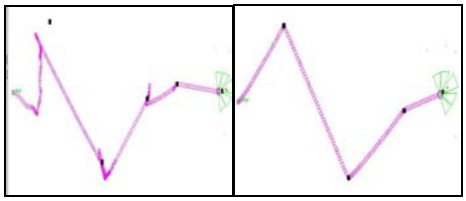


Figure 4: Trajectory of the robot during and after training.

4.2 The «obstacle avoidance» Behaviour

For this behaviour, we have determined the translation speed of the robot proportional to the distance from the frontal obstacles, with a maximum value of 350 mm/s (fig.5).

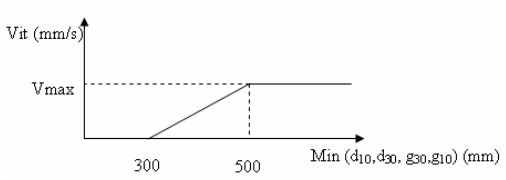


Figure 5: Translation speed Vit.

Inputs of the fuzzy controller for this behaviour are the minimal distances provided by the four sets of sonar {min(d_{90},d_{50}), min(d_{30},d_{10}), min(g_{10},g_{30}), min(g_{90},g_{50})}, with respectively three membership functions for each one (fig. 6).

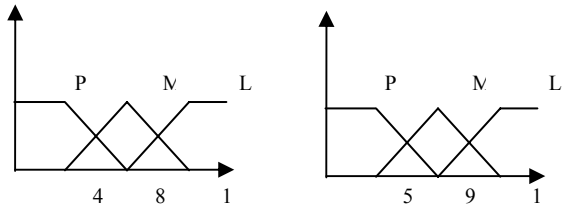


Figure 6: input membership functions for the fuzzy controller Vrot (a) :{min(d_{90},d_{50}), min(g_{90},g_{50})} (b) :{min(d_{10},d_{30}), min(g_{10},g_{30})}.

The set of the action $\boldsymbol{U}$ common to all rules consists of five actions {-20°/s, -10°/s, 0°/s, +10°/s, +20°/s}.

The reinforcement function is defined as follows:

- +1 if min {min(d_{90},d_{50}),min(d_{30},d_{10})} < min {min(g_{10},g_{30}),min(g_{90},g_{50})} & Vrot > 0
- +1 if min {min(d_{90},d_{50}),min(d_{30},d_{10})}> min {min(g_{10},g_{30}),min(g_{90},g_{50})} & Vrot < 0
- -1 otherwise

The figure (7) shows the evolution of the robot during the training phase.

On the figure (8), we represent the time evolution of the robot. It shows that the robot is able to be released from the tight corners and deadlock obstacles shape.

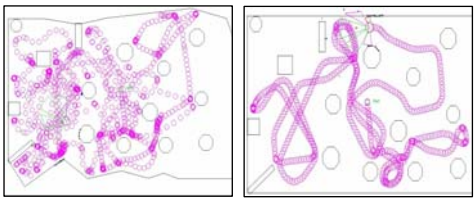


Figure 7: Trajectories of the robot during and after training.

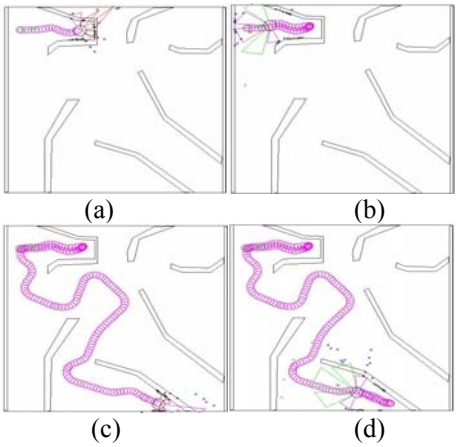


Figure 8: Time evolution of the robot after training.

4.3 Fusion of the Two Behaviours

The goal of the two elementary behaviours fusion is to allow the robot navigation in environments composed by fixed obstacles of convex or concave type and to achieve a fixed goal while ensuring its safety, witch is a fundamental point in reactive navigation.

The suggested solution consists in considering the whole input variables of the two behaviours « *goal seeking* » & « *obstacles avoidance* » associated with distributed reinforcement function by the means of a weighting coefficient between the two behaviours (0.7 for obstacles avoidance and 0.3 for the goal seeking).

The fuzzy controller input are six who are the minimal distances provided by the four sets of sonar {min(d_{90},d_{50}), min(d_{30},d_{10}), min(g_{10},g_{30}), min(g_{90},g_{50})}, with respectively three membership functions for the side sets and two membership functions for the frontal sets, to which we add the input θ_{Rb} with three membership functions, and ρ_b with two membership functions. The rules base

thus consists of 216 fuzzy rules. The translation speed of the robot is proportional to the distance from the frontal obstacles. The FACL algorithm parameters are slightly modified as follows: θ=20 & sp =0.9

Figure (9) represents the type of trajectory obtained during the training phase. The robot manages to avoid the obstacles and to achieve the goal assigned in environments encumbered by maintaining a reasonable distance between the obstacle and its with side dimensions. Figure (10) illustrates the satisfying behaviour of the robot after training. The various trajectories obtained for the same environment and the same arrival and starting points are due primarily to the problems of the perception of the environment by the robot and are related to the phenomena of sonar readings, these differences confirm at the same time the effectiveness of the training algorithm.

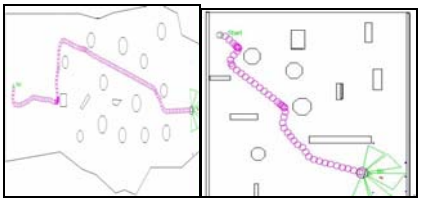


Figure 9: Trajectories of the robot in training phase.

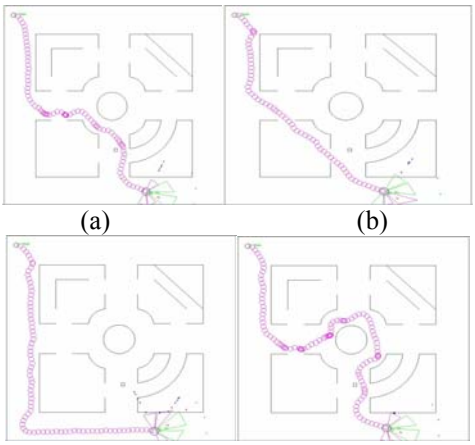


Figure 10: Various trajectories of the robot after training.

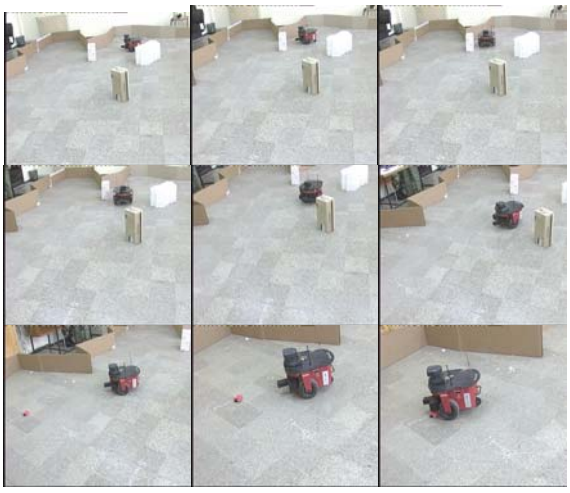


Figure 11: Trajectory of the real robot after training.

Figure (11) illustrates the satisfying behaviour of the real robot which evolves/moves in an unknown environment and which manages to achieve the fixed goal.

5 CONCLUSION

FACL Algorithm makes it possible to introduce generalization into the space of the states and the actions, and a Sugeno type order zero SIF conclusions adaptation incrementally, and this only by the means of the interactions between the apprentice and his environment. The reinforcement function constitutes the measure of the performance of the required behaviour solution. Also, fusion of the behaviours « goal seeking» & « obstacle avoidance » is presented by using a combined reinforcement function. The simulation and experimentation results in various environments are satisfactory.

REFERENCES

Babvey 03 S. Babvey, O. Momtahan, M. R. Meybodi, " Multi Mobile Robot Using Distributed Value function Reinforcement Learning", IEEE, International Conference on Robotics &Automation, September 2003.

Baird 95 L.C. Baird, "A Residual Algorithms: Reinforcement Learning with Function Approximation", Proceedings of the Twelfth International Conference on Machine Learning, 1995.

Beom 95 H.R. Beom, H. S. Cho, " A Sensor-Based Navigation for a Mobile Robot Using Fuzzy Logic

and Reinforcement Learning ", IEEE, Transactions on Systems Man and Cybernetics , vol.25, NO.3 ,pp.464-477, March 1995.

Faria 00 G. Faria, R. A.F Romero "Incorporating Fuzzy Logic to Reinforcement Learning", IEEE, pp.847-852 ,Brazil,2000.

Fujii 98 T. Fujii, Y. Arai , H. Asama, I. Endo, "Multilayered Rienforcement Learning For Complicated collision Avoidance problems", Proceedings of IEEE, International Conference on Robotics &Automation, pp.2186-2191,May 1998.

Fukuda 95 T. Fukuda, Y. Hasegawa, K. Shimojima, F. Saito,"Reinforcement Learning Method For Generating Fuzzy Controller", Department of Micro system Engineering, Nagoya University, pp.273-278, Japan, IEEE, 1995.

Glorennec 00 P. Y. Glorennec, "Reinforcement Learning: An Overview", INSA de Rennes ESIT'2000, Aachen, pp.17-35, Germany, September 2000.

Jouffe 97 L. Jouffe, "Apprentissage de Système D'inférence Floue par des Méthodes de Renforcement", Thèse de Doctorat, IRISA, Université de Rennes I, Juin 1997.

Konolige 02a K. G. Konolige 'Saphira Référence', Edition Doxygen, Novembre 2002.

Konolige 02b K. G. Konolige 'Saphira Robot Control Architecture', Edition SRI International, Avril, 2002

Smart 02 W. D. Smart, L. P. Kaelbling , " Effective Reinforcement Learning For Mobile Robots", Proceedings of the IEEE, International Conference on Robotics &Automation, pp.3404-3410, May 2002.

Sutton 98 R.S. Sutton, A. Barto, "Reinforcement Learning", Bradford Book, 1998.

EFFECTIVE GENETIC OPERATORS OF COOPERATIVE GENETIC ALGORITHM FOR NURSE SCHEDULING

Makoto Ohki, Shin-ya Uneme, Shigeto Hayashi and Masaaki Ohkita
Department of Electric and Electronic Engineering, Faculty of Engineering, Tottori University
mohki@ele.tottori-u.ac.jp

Keywords: Genetic Algorithm, Cooperative GA, Mutation Operator, Virus Operator, Nurse Scheduling, Scheduling Problem.

Abstract: This paper proposes effective genetic operators for cooperative genetic algorithm (GA) to solve a nurse scheduling problem. A clinical director of a medical department makes a duty schedule of all nurses of the department every month. Such the scheduling is very complex task. It takes one or two weeks to create the nurse schedule even by a veteran director. In conventional ways using the cooperative GA, a crossover operator is only employed for the optimization, because it does not lose consistency between chromosomes. We propose a mutation operator and a virus operator for the cooperative GA, which does not lose consistency of the nurse schedule. The cooperative GA with these new operators has brought a surprisingly good result, it has never been brought by the conventional algorithm.

1 INTRODUCTION

In hospitals, 15-30 nurses are working in a medical section such as the internal medicine department or the pediatrics department. A clinical director of the department constitutes a schedule of all the nurses every month. Constituting the nurse schedule (Goto, 1993, Ikegami 2001), or the nurse scheduling, is very complex task. In our investigation, even a veteran director needs one or two weeks for the nurse scheduling. To reduce such the problem, a computer software creating the nurse schedule is developed, and it is recentlly used at many hospitals. Those software must find the better schedule in the shorter time.

In conventional ways (Goto, 1993, Kawanaka 2002, Inoue, 2002, Itoga, 2003) using the cooperative genetic algorithm (CGA), a crossover operator is only employed for the optimization, because it does not lose consistency between chromosomes. Theirfore, the conventional technique optimizes the nurse schedule at slow speed, and it is difficult to acquire a good solution.

We propose mutation and virus operators for the CGA, which do not lose consistency of the nurse schedule. The CGA with these new operators has brought a surprisingly good result, it has never been brought by the conventional algorithm.

2 NURSE SCHEDULING BY CGA WITH CROSSOVER

In the nurse scheduling, we have to consider many requirements, for example, duty load of each nurse, fairness of assignment of day time and night duty, intensiveness of night duty and so on. These requirements are implemented by twelve penalty functions in this paper. These twelve penalty functions, F_1-F_{12}, are classified into four partial penalty groups, G_1-G_4. Finally, we define an total penalty function, E, and the partial penalty functions, G_1-G_4, as follows,

$$E(\alpha) = \sum_k {G_k}^2 , \tag{1}$$

$$G_1 = \sum_i \left(h_{11} F_{1i} + h_{12} F_{4i} + h_{13} F_{5i} \right), \tag{2}$$

$$G_2 = \sum_i \left(h_{21} F_{2i} + h_{22} F_{3i} + h_{23} F_{6i} \right), \tag{3}$$

$$G_3 = \sum_j \left(h_{31} F_{7j} + h_{32} F_{8j} + h_{33} F_{9j} \right), \tag{4}$$

and

$$G_4 = \sum_j \left(h_{41} F_{10j} + h_{42} F_{11j} + h_{43} F_{12j} \right), \tag{5}$$

where h_{pq} (p=1,2,3 or 4, q=1,2 or 3) denotes energy coefficients as follows,

h_{11}=0.2, h_{12}=0.4, h_{13}=0.4, h_{21}=0.2, h_{22}=0.2, h_{23}=0.4,

h_{31}=0.1, h_{32}=0.1, h_{33}=0.1, h_{41}=0.7, h_{42}=0.7, h_{43}=0.7. Each nurse schedule of one month is evaluated by the following penalty function,

$$H_i = h_{11}F_{1i} + h_{12}F_{4i} + h_{13}F_{5i} + h_{21}F_{2i} + h_{22}F_{3i} + h_{23}F_{6i} \quad (6)$$

In the cooperative GA, the population represents the whole nurse schedule of one month. Each individual is coded in a chromosome which shows one-month schedule of one nurse. The individual consists of the sequence of the duty symbols as shown in Fig.1. The duty sequence consists of thirty fields, since one month includes thirty days in this practical example. Each individual expresses one-month schedule of the nurse n_i. There are not two or more individuals including the same nurse's schedule in the population. An example of the population is shown in Fig.2.

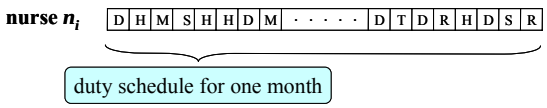


Figure 1: An individual coded in chromosome giving duty schedule of one nurse for one month.

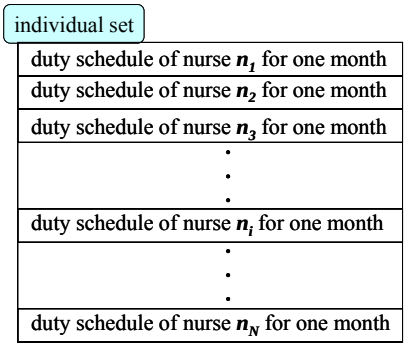


Figure 2: Population including one-month schedules of each nurse.

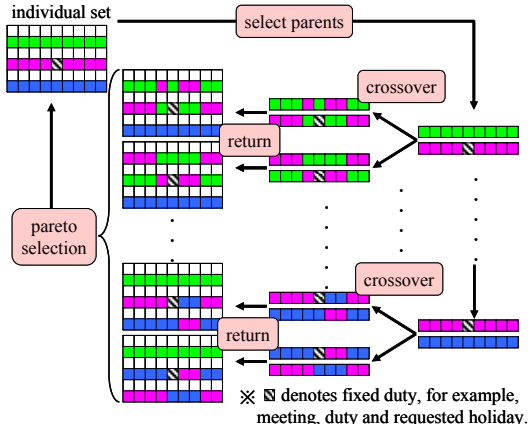


Figure 3: Optimization cycle of the CGA with the crossover operator.

Initially, the population is randomly generated as satisfying the necessary number of nurses on every duty at every date. In this paper, the necessary number of nurses is specified as ten, six and five for day time duty on a week day, Saturday and Sunday respectively, three for a semi night duty and three for night duty respectively. An overview of the crossover operator is shown by Fig.3. First, two hundreds pairs of the individual are selected as a parent for the crossover. One of the pair is selected from the population under the roulette selection manner. The roulette selection gives an individual with higher energy, H_i. This manner tends to select the worse individual. Another one of the pair is randomly selected from the population. Exchanging parts of the individuals divided at two crossover points, two new pairs of individual are generated as children. This exchanging process does not exchange the fixed duties, such as the meeting, the training and the requested holiday. These child pairs are temporally returned to the original nurse position respectively. The temporal population with the children is performed by the energy function, E. After all the pairs of individual have been performed, one pair giving the smallest energy is selected for the next generation.

3 NEW GENETIC OPERATORS

In this paper, two new genetic operators are proposed to accelerate the nurse scheduling by CGA. One of them is a mutation operator. In the conventional way, it is considered that the mutation loses the consistency of the population and does not work effectively. However, the mutation is a very strong genetic operator to widely search in the solution space. In this paper, new effective mutation operator is proposed for the cooperative GA which does not lose the consistency of the schedule. The aim of mutation is to bring small change into the population. However if one of the duty fields is randomly changed, the whole schedule become meaningless thing, which is very hard to recover by a genetic operation. Therefore, the mutation operator must preserve the number of nurses in every duty at all date. The basic operation of the mutation is shown in Fig.4. One of dates is randomly selected. Two nurses at the same date are decided. Finally, the duties of these two positions are exchanged. If one of the selected duties is the fixed duty, another nurse

is randomly selected again. The basic mutation operation is applied to CGA as shown in Fig.5.

We also propose a virus operator as another new technique. When a virus in immunology is infected within a cell, it overwrites in a part of gene forcibly. The virus copies own gene pattern in a genetic part of the cell in many cases. Our virus operator simulates this work. An aim of the virus operator is to replace some individuals with a good thing forcibly when the optimization is stagnant.

In normal optimization cycle, the CGA searches by using the crossover and the mutation operators and preserves the best performing individuals after the crossover. When the mutation operator is executed G_V times, the virus operator is applied instead of the mutation operator. One of individual who gives a bad penalty is selected by using a roulette selection manner. The virus operator overwrites the best performed individual onto the selected individual as shown in Fig.6.

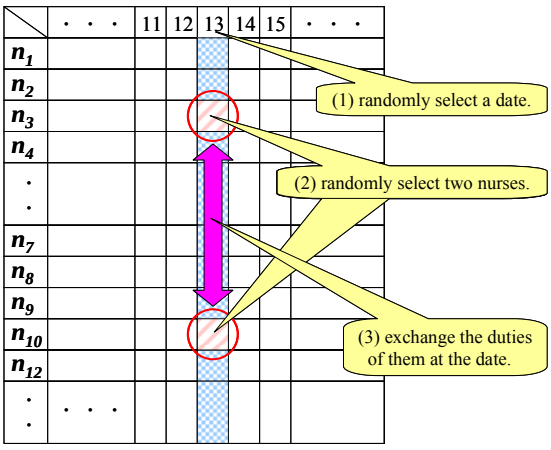


Figure 4: Primitive operation of the mutation operator.

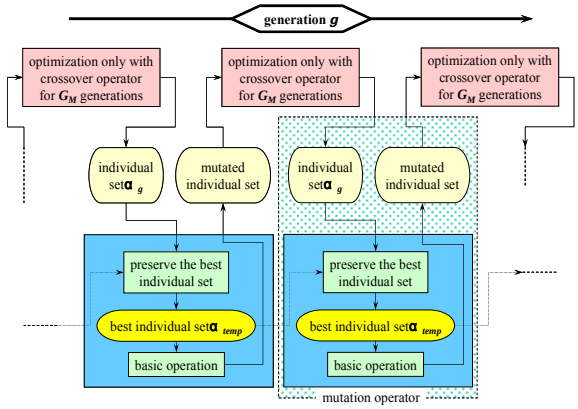


Figure 5: Mutation operator executed by a period for G_M generations.

4 COMPARISON OF GENETIC OPERATORS

We have tried three types of the CGA in this paper, the CGA only with the crossover operator, the CGA with the crossover and the mutation operators and the CGA with those three operators. An optimization of each algorithm is performed for One hundred thousands generations. It takes about ten minutes for one optimization. Ten times of trials are carried out under each condition.

Firstly, a difference of the mutation period is examined. As shown by Fig.7, the mutation period, G_M, is tried to change from fifty to two thousands generations. We have decided that the mutation period should be set at 150. Average value of the energy function for ten times trials is shown in Fig.8. The mutation with a period at less than 200 generation effectively works.

We have examined the virus operator. Different viral infection frequencies are examined as shown by Fig.9. The viral infection frequency is able to be set in a considerably wide range.

We have investigated the optimization with the virus operator in detail. After twenty or thirty thousands generations, penalty function, F_1, has big value, and the value of all other penalty functions is comparatively small. Then we have modified a part of the virus operator: when the best individual is preserved after the crossover operation, the following penalty function is applied,

$$H_i = h_{12}F_{4i} + h_{13}F_{5i} + h_{21}F_{2i} + h_{22}F_{3i} + h_{23}F_{6i}. \quad (7)$$

As shown by Fig.10, the modified virus operator gives good results with any virus frequency.

The modified virus operator does not effectively work after thirty thousands generation as shown in

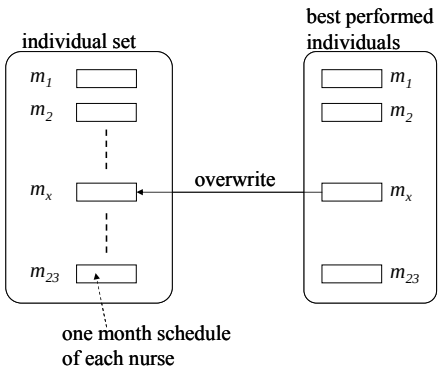


Figure 6: The virus operator forcibly overwrites an one-month schedule of one of nurses selected in the roulette selection manner, when the optimization is stagnant.

Fig.11. In contrast, the mutation operator slowly searches. Then we apply the modified virus operator until thirty thousands generations. After that, the mutation operator is only applied to the CGA with the crossover. As shown in Fig.11, the rearranged technique gives the best result.

5 CONCLUSION

We have proposed two new genetic operators for the CGA applied to the nurse scheduling. The only one search technique of the conventional CGA is the crossover operator, because it dose not lose the consistency of the nurse schedule. In contrast, the new techniques which we proposed in this paper give good results without losing the consistency. By means of these new techniques, the optimization of the nurse schedule has been accelerated.

REFERENCES

Goto, T., Aze, H., Yamagishi, M., Hirota, M. and Fujii, S. 1993, *Application of GA, Neural Network and AI to Planning Problems*, NHK Technical report, No.144.

Ikegami, A. 2001, *Algorithms for Nurse Scheduling*, Proc. of 11th Intelligent System Symposium.

Kawanaka, S., Yamamoto Y., Yoshikawa D., Shinogi T. and Tsuruoka N., 2002, *Automatic Generation of Nurse Scheduling Table Using Genetic Algorithm*, Trans. on IEE Japan, Vol.122-C.

Inoue, T., Furuhashi, T., Maeda H. and Takabane, M., 2002, *A Study on Interactive Nurse Scheduling Support System Using Bacterial Evolutionary Algorithm Enegine*, Trans. on IEE Japan, Vol.122-C.

Itoga, T., Taniguchi N., Hoshino Y. and Kamei K., 2003, *An Improvement on Search Efficiency of Cooperative GA and Application on Nurse Scheduling Problem*, Proc. of 12th Intelligent System Symposium.

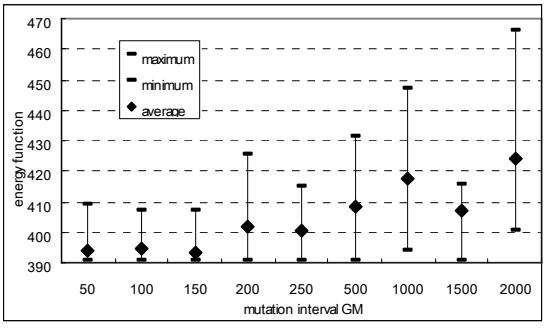


Figure 7: Comparison of the mutation period.

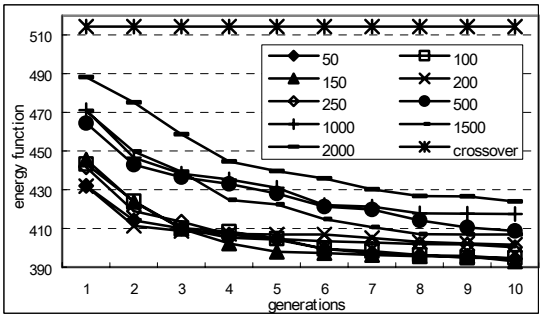


Figure 8: Average value of the energy function for ten times trials. The unit of the horizontal line is ten thousand.

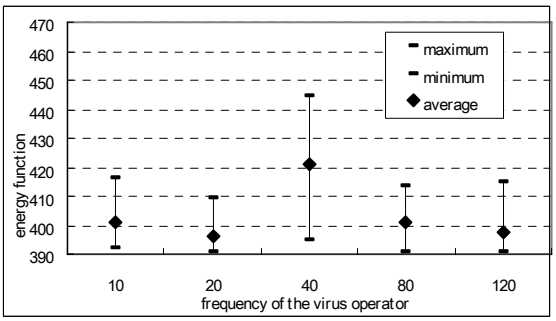


Figure 9: Comparison of the frequency of the virus operator.

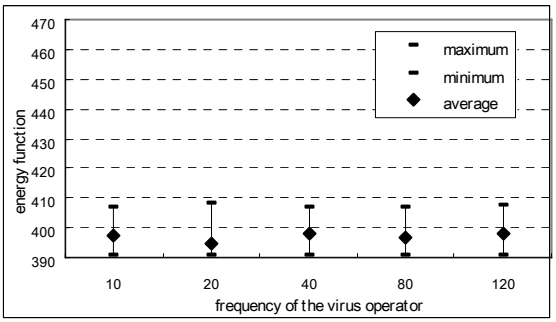


Figure 10: Comparison of the frequency of the modified virus operator.

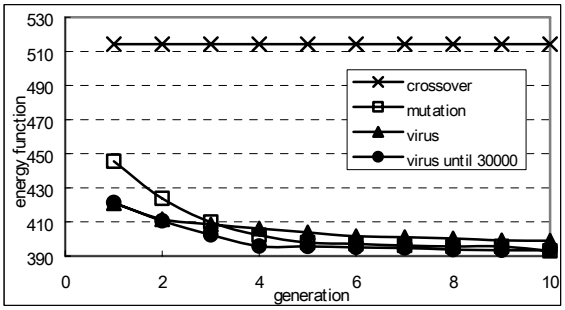


Figure 11: Comparison of the optimization process among CGA only with the crossover operator, CGA with the mutation operator, CGA with the modified virus operator and CGA applied with the modified virus operator until 30000 generations. The unit of the horizontal line is ten thousand.

OBJECT LIST CONTROLLED PROCESS DATA SYSTEM

Anton Scheibelmasser
CAMPUS 02, University of Applied Sciences, Automation Technology, Körblergasse 126, 8021 Graz, Austria
anton.scheibelmasser@campus02.at

Bernd Eichberger
Graz University of Technology, Department of Electronics, Inffeldgasse 12, 8010 Graz, Austria
bernd.eichberger@tugraz.at

Keywords: Object oriented design, process control system, linked object list.

Abstract: The appropriate design of a system is one of the essential topics at the beginning of a new development project. According to the intended purpose of a device the first step is to model the system in order to get a structure for the implementation of the required features. In general the implementation of the system requirements is split in hardware parts and tasks which are done in software. In case of the hardware design the solutions for the challenges are mostly clear and supported by fundamentals of e.g. digital logic laws and several design methods. If we think of the software part a lot of problems have to be solved without such clear fundamentals. Object oriented design is one of the paradigms which promise a way for designing stable and reliable software. A problem arises in this context if the used microprocessor platform is not supported with a compiler for an object oriented programming language. In this case only the system modelling could be done in terms of software objects and their relations, the implementation has to be done in a procedural language. The following article is based on research work done in the development of a modular process data system. Based on a sequential main program and interrupt driven hardware interfaces, a software implementation without an operating system was implemented. By means of special software structure called Linked Object List, object oriented design was implemented with the procedural language "C". Due to this design a reusable and flexible system was achieved which enables a high degree of flexibility concerning the hardware configuration and system customization at the user site.

1 INTRODUCTION

Controlling a process in an industrial environment requires the measurement of the relevant process quantities. The use of process lines helps to master the often very complex system structure. In order to master this complexity, distributed small process controllers operate through a network with a host computer located at a central point of the plant. The software design for such remote controllers requires a high degree of flexibility to handle the various hardware options. This can be achieved by using intelligent software algorithms. Depending on the specific hardware interfaces and the process line, an adaptation in the field by means of parameterisation or configuration can be performed easily.

2 DESIGN CONSIDERATIONS

Based on the generic user requirement specification, at a first step the system requirements were modelled in hardware and software parts.

In contrast to the hardware design, the software structure is not so simply derived from the specifications. On the one hand standard applications which are available for microcontroller designs and digital circuits like LCD interfaces or PC-Card implementations are not available, on the other hand basic conditions of the system (e.g. operating system, programming language, development platform) have to be defined.

One paradigm in the field of software development is the usage of object-oriented methods to analyze, model and implement the software requirements. Object-oriented design (Stroustrup, 1991) requires the modeling of the problem by means of a data structure called Class. The goals of

such a design are to hide the complexity (abstraction) and to protect the data effectively. If we model our system according to these principles we can structure our software into different classes (objects) and their interfaces (methods). A system build up in this way supports the developer with a lot of advantages concerning reuse, maintenance and stability.

A problem occurs if the used microprocessor platform is not supported by a compiler for an object-oriented language. For many controller platforms only a C-compiler is available but no C++-compiler. A typical solution for this problem is a trade-off between the economical/technical decision for a microcontroller and the software development restrictions concerning the object-oriented language. But the missing of an object-oriented language does not imply that we have to develop our software without object-oriented principles. A good practice is to analyze and to model the software requirements with object oriented tools and methods (e.g. UML -diagrams). The implementation is done later in a procedural language with a few restrictions. For the implementation of object-oriented designs with the procedural language C, a trade-off can be achieved by replacing the C++-class with the C-structure. The methods (member functions) can be replaced as function-pointers in such a structure. Protection mechanisms are possible by putting such a C-structure into a single C-File and to control the visibility of the attributes (member variables) by means of an H-File (header file). Therefore access to the members of the C-structure is only possible by means of dedicated member functions (e.g. get(), set()). Another aspect is the instantiating of classes as objects. This feature can be implemented with arrays of such structures.

3 SOFTWARE DESIGN OF THE PROCESS DATA SYSTEM

To analyze and model the requirements of the process data system, UML diagrams (Fowler, 1999) were taken to get an overview of the system. As a first result of the requirement analysis the use case diagram (figure 3) was generated. A central part of the system represents the application program which is optionally extendable with programs on a memory card (update, user applications). Depending on the configuration of the system, process variables are measured, calculated and distributed to the host computer or to the graphical display.

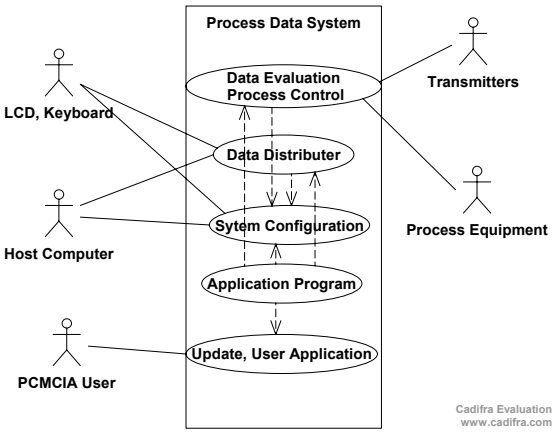


Figure 1: Use Case Diagram of the Process Data System.

The configuration process is one of the central tasks in the system. Program handling, data measurement and distribution as well as the handling of the transmitter and process environment options should be handled according to the configuration data. To manage the versatility of transmitters, process lines and process equipments a generic algorithm was necessary.

3.1 Process Data Object

In a next step of the software development process the modelling of the system by means of classes (objects) has to be done. In order to handle the demanded versatility, a Data Object Class was introduced. Every process parameter, variable or constant should be an instance (object) of this class. A few attributes of the object control the behaviour of the process value in the device. For every component of the system one attribute of the class is responsible for controlling the data handling. For instance, if the process variable `density` should be measured and shown on the display a member variable `HARDWARE-MASK` and `DISPLAY-MASK` has to be set appropriately. Depending on this object definition, every module in the software has to be written generically. This implies that the respective program part has to evaluate the attributes of the data object before handling the process value. Process Data Objects are very flexible concerning their handling in terms of definition. Hence user specific object definitions are able to extend the constant object definitions of the main system. Therefore a flexible way of customizing the default system is possible. For instance, the system software can be extended by means of an external data memory which stores additional Process Data Objects.

A Device Builder component, implemented by means of a state machine, prepares the basic state values for the Process Objects. The state function of the Device Builder interacts with several components in a sequential way. Starting with the invocation of the measurement tasks, the application program, the visualisation and the host communication, interfaces of the respective components are used. The Process Object component plays a central role in the system. Every component has to evaluate the attributes of the handled process data objects to get information about the intended use, the type and the specific handling of the values.

3.2 Implementation Issues

Based on the system modelling and the demanded requirements, a few basic conditions have to be defined. One decision concerns the usage of an operating system, another one the selection of the appropriate programming language.

In order to meet economical aspects and based on experiences from former projects, a decision was made not to use an operating system. Supported by the fact that the given requirement does not need operating system support like e.g. file system, network stack or multitasking, lower memory requirements have been achieved. The real time aspect in this context has already been solved in a former project with the implementation of the measurement tasks based on interrupt driven transmitter routines.

The second question concerning the programming language was solved with the selection of the used microprocessor for economical/technical reasons. As the development platform of this microcontroller (Keil, 2007) supports only the procedural language "C", the object oriented design has to be implemented as a trade-off in a procedural way. Based on these decisions the main frame of the software was implemented as a sequential main()-procedure supported with interrupt driven routines for interfacing the process equipment (e.g. transmitters). To support distributed development and reuse for further projects, the graphical system for the LCD was implemented as separated procedures and linked by means of a software-interrupt (Graphic-BIOS). One of the central points in this context was the implementation of the Process Data Object class by means of arrays of C-structures.

3.3 Procedural Object Class Implementation

According to the language "C" the Process Data Object was implemented as C-structures. As there are a lot Process Data Objects in the system, a constant array of such structures was defined. Instead of instantiating objects, a constant array of structures was defined which represents the main definitions of the system's behaviour. Every structure incorporates a name, a type and behaviour attributes for the process data. Based on the type of data a method, implemented e.g. as a function pointer, is responsible to access the respective value. Based on the requirement to extend the software at the customer site special process data objects were introduced with the object type `OBJ-LINK`, and `OBJ-LAST`. Due to this definition it was possible to connect several Process Data Object arrays in a way which is called Linked-List. By means of this inked object list it was possible to extend the basic definitions of the system behaviour with customer definitions at compile time or later in the field by means of structures located at the mobile memory card. A central procedure was introduced to search for an object in this linked list.

Based on the object's type in the structure this procedure was able to evaluate the `OBJ-LINK` type and to connect several arrays of structures. Depending on the start point of the search (e.g. system memory, customer memory, PC-Card card) a certain priority was achieved which is usable to override (customize) basic system definition.

For instance with the introduction of new object definitions on the memory card, new process data readings or menu items are available in the process data system. The following code fragments are intended to give an impression of the structure definition and the outlook of the object list.

```
typedef struct
   {
      NAME name;
      union
      {
         unsigned long c_li;
         NVPARA * p;
         FUNC_adr funct;
         unsigned long * i;
         float * df;
         char * str;
         NAME * nstr;
      } data;
      char d_name[MAX_DLEN+1];
      char d_unit[MAX_DLEN+1];
      char d_format[MAX_DLEN+1];
      char hierarchy[MAX_BRANCH+1];
      unsigned char hardware_mask;
```

```
        unsigned char handle_attrib;
        unsigned char obj_type;
    } OBJECT;

const OBJECT Test_Object_Table[] =
    {
      {
         "Main-Menu", NULL,
         "","","",
         {0,0,0,0,0,0,0,1},
         TM_NONE,
         D_MENU,
         OBJ_MENU },
      {

    "Temp",(float*)&MEAS_results.pte,
    " Temp."," [ \xf8""C ]","%8.2f",
    {0,0,1,5,5,6,ALL,13},
    TM_NONE,
    D_MENU | A_ASSIGN,
    OBJ_FLOAT }
    // a lot of additional objects …
  }
```

As shown in this structure and their definitions, not only attributes concerning the visualisation and the hardware are available but also definitions for the format and the hierarchies in menu trees. The Device Builder component was implemented as a state machine in the main() -procedure of the program. Depending on the state function (Data Visualisation, Menu Selection, Parameter Edit or Adjustment, System Configuration) criteria are set which influence the search procedure (e.g. getobject()). So if every component uses the Linked Object List with the process data definitions, a pure generic system is established which enables a high degree of dynamic concerning the implementation of hardware- or software options.

4 RELATED WORKS

Innumerable publications describe different methods of achieving reusable object-oriented software designs. One of these methods is called Design Patterns (Gamma, 1994). The idea is to use elegant and proven solutions for dedicated problems to model the core of the software at the start of the design. The method of mastering the complexity and the high dynamical efforts of hardware/software options in the system by means of Object List controlled procedures is similar but not identical to a design pattern called Command (Gamma, 1994). This pattern is described as an encapsulation of a request in an object. By means of this pattern clients could issue request to objects without knowing anything about the operation requested. In this context process data objects can be seen as such requests to other clients (components of the system). In this work the idea was modified and extended with features of linked lists to handle the demanded customization. A further idea was the implementation of such a pattern in a procedural language.

5 CONCLUSIONS

In the development of a new process data system high requirements concerning the implementation of different hardware options and customization features were fulfilled. The introduction of an "Object List Control" structure has significantly enhanced the stability, robustness, maintenance and extensibility of the software structure. Depending on this structure a high degree of flexibility was achieved and the requirements concerning customization at the vendor's site and in the field were met. The object list structure has proved its capability to solve the problem of customization by binding different device configurations (object lists) at compile or at runtime. Based on this list control new device features can be added by means of mobile data cards. From our point of view this method is advisable if there are a lot of configurable options defined in the system or in case of weak or partial requirement specifications. In both cases a high degree of flexibility in software design is needed. The open "Linked Object List" is a generic approach which allows upgrading the system behaviour without recompiling the whole software.

REFERENCES

Stroustrup B., 1991. *What is "Object-Oriented Programming?".* AT&T Bell Laboratories Murray Hill, New Jersey 07974.

Fowler M.,*UML Distilled, Second Edition*, 1999, Addison-Wesley

CADIFRA, *UML Editor, 2005,* www.cadifra.com

KEIL GmbH, 2007, *C166 Development Tools,* www.keil.com

Gamma E., Helm R., Johnson R., Vlissides J., *Design Patterns,* 1994, Addison-Wesley

OPTIMIZATION MODEL AND DSS FOR MAXIMUM RESOLUTION DICHOTOMIES

James K. Ho
Department of Information and Decision Sciences, University of Illinois at Chicago, IL 60607, USA
jimho@uic.edu

Sydney C. K. Chu
Department of Mathematics, University of Hong Kong, Pokfulam Road, Hong Kong SAR, China
schu@hku.hk

S. S. Lam
School of Business and Administration, The Open University of Hong Kong, Hong Kong SAR, China
sslam@ouhk.edu.hk

Keywords: Data mining, decision support system, maximum resolution topology, multi-attribute dichotomy, goal programming, optimization modelling.

Abstract: A topological model is presented for complex data sets in which the attributes can be cast into a dichotomy. It is shown that the relative dominance of the two parts in such a dichotomy can be measured by the corresponding areas in its star plot. An optimization model is proposed to maximize the resolution of such a measure by choice of configuration of the attributes, as well as the angles among them. The approach is illustrated with the case of online auction markets, where there is a buyer-seller dichotomy as to whether conditions are favourable to buyers or sellers. An implementation of the methodology in a spreadsheet based DSS is demonstrated. Its ease of use is promising for diverse applications.

1 INTRODUCTION

A topological model for a high dimensional data set is a simultaneous graphical display of all its relevant attributes, which provides a geometrical shape as a descriptive, visual statistics of the underlying construct engendering the data. In particular, when various dimensions can be identified to form a multi-attribute dichotomy, the area spanned by the two halves of the topological model can be used as a measure of the relative dominance of the two parts of the dichotomy. Using a reference subset of prejudged cases, the configuration of the dimensions and the angles among them can be optimized in a Goal Programming (Scniederjans, 1995) model for a topology that maximizes the resolution of such dichotomies. Applications abound in diverse fields, including diffusion of innovation (Ho, 2005), investment climate and business environment (Ho, 2006a), marketing research and customer relations management (Ho, 2006b), and medical diagnostics. The implementation of the optimization model as an easy-to-use, spreadsheet based DSS is described. It is illustrated by the case of topological analysis of online auction markets (Ho, 2004) where it is of interest to discern whether particular markets are favourable to buyers or sellers.

2 TOPOLOGICAL ANALYSIS

Visualization has been a fast developing approach in data-mining (Hoffman and Grinstein, 2001) in which graphical models are constructed to provide visual cues for pattern recognition and knowledge discovery from complex data. In the study of financial markets (stock and commodity), the dimension of interest is primarily prices, or the fluctuation thereof. Complexity arises from the large number of instruments involved. The best known examples of visualization models for stock markets are based on the tree-map method (Shneiderman,

1992), and the minimum-spanning-tree method (Vandewalle et al, 2001). For auction markets, the game-theoretic dynamics itself gives rise to higher dimensional complexity. And with online auctions removing conventional constraints on time and space, their activities and impact on e-commerce can only be expected to grow exponentially. In this regard, the availability of operational data from eBay.com presents unprecedented challenges and opportunities for insight into online auction markets. In (Ho, 2004), twelve dimensions (i.e. attributes) are identified as follows.

1. NET ACTIVITY (auctions with bids)
2. PARTICIPATION (average number of bids per auction)
3. SELLER DIVERSITY (distribution of offers)
4. SELLER EXPERIENCE (distribution of sellers' ratings)
5. MATCHING (auctions ending with a single bid)
6. SNIPING (last minute winning bids)
7. RETAILING (auctions ending with the Buy-It-Now option)
8. BUYER DIVERSITY (distribution of bidder participation)
9. BUYER EXPERIENCE (distribution of buyers' ratings)
10. DUELING (evidence of competitive bidding)
11. STASHING (evidence of stock-piling)
12. PROXY (use of proxy bidding as evidence of true valuation)

Our topological model is based on the star plot for displaying multivariate data with an arbitrary number of dimensions (Chambers et al, 1983). Each data point is represented as a star-shaped figure (or glyph) with one ray for each dimension. As the resulting shapes depend on the configuration of the dimensions, we further analyse the observations along the dimensions identified above in an effort to present a visual model of the shape of online auction markets.

To discern whether particular market conditions are favourable to buyers or sellers, we divide the dimensions into a buyer-seller dichotomy as shown in Figure 1 where buyer dimensions (SELLER DIVERSITY, SELLER EXPERIENCE, MATCHING, SNIPING, RETAILING) are grouped to the right, and seller dimensions (BUYER DIVERSITY, BUYER EXPERIENCE, DUELING, STASHING, PROXY) are grouped to the left. The other dimensions (NET ACTIVITY, PARTICIPATION) are neutral and mapped to the vertical axis.

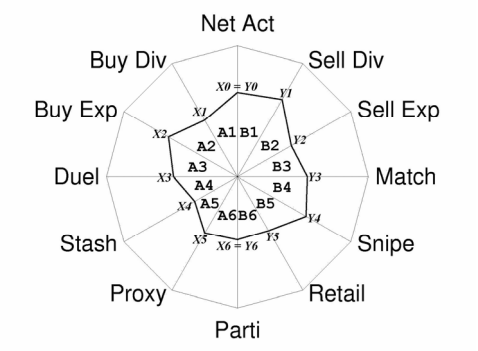


Figure 1: Topological model of online auction market.

3 MAXIMUM RESOLUTION TOPOLOGY

In general, a multi-attribute dichotomy is any multi-dimensional dataset in which the dimensions can be partitioned into two groups, each contributing to one part of the dichotomy. Given the star glyph of a multi-attribute dichotomy, as exemplified in Figure 1, it will be both visually and intuitively appealing if the areas covered by the two parts can be used as a meaningful aggregate measure of their relative dominance. A larger area on the left side of the glyph means dominance by the left part, and vice versa. In the case of online auction markets, this asymmetry can be interpreted as market conditions being advantageous to either buyers or sellers. In mathematical terms, the aggregate value function takes the form of the sum of pair-wise products of adjacent attributes: $V(X_1, \ldots, X_n) = C \Sigma X_i X_j$; where attributes i and j are adjacent; X_i is the value of attribute i, for i = 1, …, n; and C is some scaling constant.

The concept of using the area of the parts of a dichotomy as an aggregate measure of their relative dominance is plausible, since increasing value of an attribute contributes positively to its designated part, as well as the latter's area in the glyph. However, it must be refined to realize its potential, which arises from the degrees of freedom allowed by the topology of the glyph, namely, the configuration of the attributes, and the angles between adjacent pairs thereof. For any given arrangement of the attributes, the standard star plot produces a glyph along symmetrically spaced radial axes. Variations from this symmetry imply a feasible set of shapes and areas, which along with permutations of the configuration, offer the choice of topologies that may suit further criteria for a meaningful aggregate

measure function. In particular, we use a diverse subset of the data instances in an optimization model to derive a topology with maximum resolution in discerning dominance with respect to the reference subset (Ho and Chu, 2005).

To this end, the first step is to render the glyph unit free by normalizing the data on each dimension to the unit interval [0, 1]. The second step is to render the glyph context free by harmonizing the dimensions as follows. For each attribute, the quartiles for the values in the entire dataset are computed. A spline function (Cline, 1974) is constructed to map these quartiles into the [0.25, 0.5, 0.75] points of the unit interval. This way, a hypothetical data instance with all attributes at mean values of the dataset will assume the shape of a symmetrical polygon with vertices at the mid-point of each radial axis. In this frame of reference, all shapes and sizes are relative to this generic "average" glyph, and free of either units or specific context of the attributes. For our exploratory work, simple second-order (piecewise linear) splines are used.

3.1 Dichotic Dominance with respect to Reference Subsets

Next, to determine an optimal topology, we use the concept of a reference subset of the data instances to help define dichotic dominance. This concept is best explained in a medical scenario. Suppose a certain disease is monitored by a number of symptoms and tests, with a dichotic prognosis of "life" or "death". Judging from the combination of data for any particular case, it may be difficult to predict. A reference subset is a collection of non-trivial, non-obvious cases with known outcomes, namely life or death. In our exploratory analysis of online auction markets, there is no factual or expert judgment on whether any particular case is a "buyers" or "sellers" market. An initial collection from 34 diverse and well-established markets is used on an ad hoc basis as the reference subset. An arbitrary configuration of the attributes within each part of the dichotomy is selected with the attributes evenly spaced, as in Figure 1. This is analogous to selecting a portfolio of stocks to provide an index for a stock market. The performance of any stock can be gauged relative to the index, which may be arbitrarily chosen initially. With better knowledge of the significance of individual stocks, more useful indices can be established. By the same token, the choice of reference subsets for multi-attribute dichotomies can be adaptively refined as the study progresses.

Once an optimal topology is derived with respect to a given reference subset, any other data instance, an online auction market in our case, can be plotted and visualised as a maximum resolution dichotomy. Moreover, the total enclosed area in the plot, including both parts of the dichotomy may be used as a relative measure of the overall activity of all the attributes. We can consider this as an indicator of the "robustness" of the market. Whereas, the difference in the areas of the left and right parts of the dichotomy provides an index of dichotic dominance among market conditions favouring buyers and sellers. In our settings, a left dominance favours sellers, and a right dominance favours buyers.

3.2 A Goal Programming Optimization Model

Subject to the constraints of preserving the prejudged dominance in the reference subset of dichotomies, an optimal topology (configuration of attributes and angles between adjacent pairs) is sought that maximises the discriminating power, or *resolution*, as measured by the sum of absolute differences in left and right areas for the reference subset. Such an optimal configuration will be called a *maximum resolution topology* (MRT). For any given configuration of the attributes, maximization of the discriminating power can be formulated as a linear program (LP). However, LP produces extreme-point solutions, which may reduce some of the angles between attributes to zero, thus collapsing the glyph. To avoid such degeneration, maximization with bounded variation of the angles is modelled as a goal program (GP) in (Ho and Chu, 2005).

4 DSS FOR MRT

To facilitate the computation of a maximum resolution topology (MRT) for a given set of data from a multi-attribute dichotomy, an easy-to-use decision support system (DSS) has been built on Excel spreadsheet software. Such an MRT-DSS system has both its front end and report routine integrated in the same Excel spreadsheet workfile, into which the input data records can be placed (for example, imported from a database); and outputs of values and MRT-star plots displayed.

To find the solution, the user only needs to copy and paste the records of training data (the "reference

set") to the 'Training Data' worksheet and click the 'Solve!' item button on the 'MRT' menu. MRT-DSS will permute over all possible configurations and dynamically generate the input data for each configuration. The training data will be passed to a linear programming solver (LINGO Version 8) to find the solution based on the MRT-GP model. MRT-DSS will store the solution of each configuration on the 'Work' worksheet, as well as the best solution on the 'Best solution' worksheet. It will also keep the optimal MRT configuration and angles in the 'StarPlot' worksheet for preparing the test data for plotting.

By completing the training of MRT-DSS and obtaining the optimal configuration, the system can then be used to evaluate new cases of the dichotic model. With data copied to the 'Testing Data' worksheet, the 'Prepare StarPlot Data' item on the 'MRT' menu is selected. MRT-DSS will transpose and store the data in the 'StarPlot' worksheet. It will also compute for each test case the areas of the left (A) and right (B) parts of the dichotomy and their difference (A-B). The user can easily evaluate the test cases based on these numerical results. To visualize and further analyze a particular data record, the user can choose the 'Plot Solution' item on the 'MRT' menu to draw its StarPlot diagram under the maximum resolution topology. By inspecting and comparing records under the optimal configuration and angles in the diagrams, and by studying the left-right differentials provided by MRT-DSS, substantial topological analysis can be performed for insight into the model under study.

5 CONCLUSIONS

We presented an optimization model to derive a maximum resolution topology for complex data sets that can be cast as multi-attribute dichotomies. While we used only the buyer-seller dichotomy for online auction markets as illustration, applications have already resulted in diverse fields (Ho, 2005, 2006a, b). For future work, we expect ample innovative applications of the methodology with the help of the easy-to-use DSS.

ACKNOWLEDGEMENTS

This work is partially supported by the Hong Kong RGC Competitive Earmarked Research Grant (CERG) Award: HKU 7126/05E.

REFERENCES

Chambers, J., Cleveland, W., Kleiner, B. and Tukey, P., 1983. *Graphical Methods for Data Analysis*, Belmont, CA: Wadsworth Press.

Cline, A. K., 1974. 'Scalar- and planer-valued curve fitting using splines under tension', *Communications of the Association for Computing Machinery* 17: pp. 218–220.

Ho, J. K.,2004. 'Topological analysis of online auction markets', *International Journal of Electronic Markets* 14(3): pp. 202–213.

Ho, J. K., 2005. 'Maximum resolution dichotomy for global diffusion of the Internet', *Communications of the Association for Information Systems* 16:pp.797–809.

Ho, J. K., 2006a. 'Maximum resolution dichotomy for investment climate indicators', *International Journal of Business Environment* 1(1): pp.126–135.

Ho, J. K., 2006b. 'Maximum resolution dichotomy for customer relations management', in Zanansi, A. et al (eds.) *Data Mining VII,* WIT Press, Southampton, pp. 279–288.

Ho, J. K. and Chu, S. C. K. (2005) 'Maximum resolution topology for multi-attribute dichotomies', *Informatica* 16 (4): pp. 557–570.

Ho, J. K. and Chu, S. C. K. and Lam, S.S. (2007) 'Maximum resolution topology for online auction markets', *International Journal of Electronic Markets* 17(2) (to appear).

Hoffman, Patrick and Grinstein, George, 2001. 'A survey of visualizations for high-dimensional data mining', in Usama Fayyad et al (eds.) *Information Visualization in Data Mining and Knowledge Discovery*, Morgan Kaufmann: pp. 47–82.

Roth, Alvin E. and Ockenfels, Axel (2002) 'Last minute bidding and the rules for ending second-price auctions: theory and evidence from A natural experiment on the Internet', *American Economic Review* 92(4): pp. 1093–1103.

Scniederjans, M. J., 1995. *Goal Programming Methodology and Applications,* Kluwer publishers, Boston.

Shneiderman, B., 1992. Tree visualization with tree-maps: a 2-D space-filling approach. *ACM Transactions on Graphics* 11: pp. 92–99

Tukey, J.W., 1997. *Exploratory Data Analysis*. Addison-Wesley, Reading, MA.

Vandewalle, N., Brisbois, F., Tordoir, X., 2001. 'Non-random topology of stock markets'. *Quantitative Finance* 1: pp. 372–374.

CHANGE-POINT DETECTION WITH SUPERVISED LEARNING AND FEATURE SELECTION

Victor Eruhimov, Vladimir Martyanov, Eugene Tuv
Intel, Analysis and Control Technology, Chandler, AZ, U.S.A.
eugene.tuv@intel.com

George C. Runger
Industrial Engineering, Arizona State University, Tempe, AZ, U.S.A.
runger@asu.edu

Keywords: Data streams, ensembles, variable importance, multivariate control.

Abstract: Data streams with high dimensions are more and more common as data sets become wider. Time segments of stable system performance are often interrupted with change events. The change-point problem is to detect such changes and identify attributes that contribute to the change. Existing methods focus on detecting a single (or few) change-point in a univariate (or low-dimensional) process. We consider the important high-dimensional multivariate case with multiple change-points and without an assumed distribution. The problem is transformed to a supervised learning problem with time as the output response and the process variables as inputs. This opens the problem to a wide set of supervised learning tools. Feature selection methods are used to identify the subset of variables that change. An illustrative example illustrates the method in an important type of application.

1 INTRODUCTION

Data streams with high dimensions are more and more common as data sets become wider (with more measured attributes). A canonical example are numerous sensors (dozens to hundreds) with measurements generated from each over time. Many characteristics can be of interest from a system that generates such data, but one systemic question is whether the system has been stable over a time period, or whether one of more changes occurred. In a change-point problem, historical data from streams is reviewed retrospectively over a specified time period to identify a potential change, as well as the time of the change. This historical analysis differs from real-time monitoring where the goal is to detect a change as soon as it occurs.

Change points are of interest in areas as diverse as marketing, economics, medicine, biology, meteorology, and even geology (where the data streams represent data over depths rather than over time). In medicine, a change-point model can be used to detect whether the application of a stimulus affects the reaction of individual neurons (Belisle et al., 1998). In the study of earthquakes, it is of interest to distinguish one seismicity from another (Pievatolo and Rotondi, 2000).

Modern data streams often must handle high dimensions. A common approach is to use a multivariate control chart for process monitoring such as Hotelling's T^2 control chart (Hotelling, 1947). This is a widely-used multivariate control chart to monitor the mean vector of a process based on the Mahalanobis distance of the current data vector from a historical mean data vector. The distance measure used in T^2 incorporates the correlations among the attributes that are measured. However, because this distance measure is fundamentally based on a sum of squared deviations of the elements of the current vector, it loses sensitivity to change points that occur in only one or a few attributes among many (and result in small changes in Mahalanobis distance). More sensitive extensions were developed for real-time monitoring such as a multivariate exponentially weighted moving average control charts (MEWMA) (Lowry et al. 1992), and a multivariate cumulative sum control charts (MCUSUM) (Runger and Testik 2004). These extensions are still based on sums of squares with the previously mentioned, intrinsic limitations as the dimension increases.

The objective here is to handle the high-dimensional, complex data that is common in modern sensed systems, and still detect change point that might occur in only one (or a few) variable among hundreds. Consequently, we present a two-phased approach. In the first phase we identify the attributes responsible for the change point. With a much smaller subset of attributes to work with in the second phase, simpler methods can be used to identify the time(s) at which the change(s) occur. The first phase uses a novel transformation of the problem to one of supervised learning. Such a transformation was explored by (Li et al., 2006). The work here adds a second phase, uses a much more powerful feature selection algorithm, and provides a more challenging example. In Section 2 the change-point problem is transformed to a supervised learning problem. Section 3 discusses feature selection. Section 4 provides a realistic example.

2 CHANGE POINTS WITH SUPERVISED LEARNING

A supervised learning model requires a response or target variable for the learning. However, no obvious target is present in a change-point problem. Still, a key element of a data stream is the time attribute that provides an ordering for the measured vectors. In a stationary data stream without any change points, no relationship is expected between time and the measured attributes. Conversely, if the distribution changes over time, such change should allow for a relationship to be detected between the measured attributes and time (Li et al., 2006). Consequently, our approach is to attempt to learn a model to predict time from the measurements in the data stream

$$t = g(x_1, ..., x_p) \quad (1)$$

where t is the time of an observation vector and $g()$ is our learned model. If the time attribute can be predicted, a change in the measurement vectors is available to predict. Attributes that are scored to be important to this model are the subset of important variables. Consequently, phase one of our analysis can be completed from this model and its interrogation. Any number of change points can occur in this framework.

A more direct approach might attempt to model each attribute as a function of time such as $x_j = g(t)$ for $j = 1, 2, \ldots, p$. However, separate models do not use the relationships among the variables. A change might break the relationships between variables within a significance difference in each variable individually. Common examples in data streams depict points that are not unusual for any attribute individually, but jointly depict an important change.

Any monotonic function of time can be used as the target attribute for the learner. The identify function used here is a simple choice and other functions can be used to highlight or degrade the detection of change points in different time periods. Also, any one of many supervised learners can be applied. Our goal is to detect a subset of important variables and this is the primary purpose for our following selection.

Because we are most interested in an abrupt change in the mean of one or more attributes in the data stream it is sensible to use a supervised learner that can take advantage of such an event in the system. Furthermore, the phase one objective is to identify the important variables. Consequently, decision trees are used as the base learners because they can effectively use a mean change in only one or few predictor attributes. They also have intrinsic measures of variable importance. Ensembles of decision trees are used to improve the measure of variable importance for the phase one objective.

3 FEATURE SELECTION

If an attribute changes over time, it should be more useful to predict time than an attribute that is statistically stable. Consequently, the phase to identify changed attributes is based on a feature selection method for a supervised learner. There are several approaches such as filter, wrapper, and embedded methods. An overview of feature selection was provided by (Guyon and Elisseeff, 2003) and other other publications in the same issue. Also see (Liu and Yu, 2005). The feature selection phase needs to process hundreds of attributes and potentially detect a contribution of a few to the model to predict time. Furthermore, in the type of applications of interest here, the attributes are often related (redundant). Consequently, the effect of one attribute on the model can be masked by another. Moderate to strong interactive effects are also expected among the attributes. Consequently, a feature selection methods need high sensitivity and the ability to handle masking and interactive effects. We use a feature selection methods based on ensembles of decision trees.

Tree learner are fast, scalable, and able to handle complex interactive effects and dirty data. However, the greedy algorithm in a single tree generates an unstable model. A modest change to the input data can make a large change to the model. Supervised ensemble methods construct a set of simple models (called

base learners) and use their vote to predict new data. Numerous empirical studies confirm that ensemble methods often outperform any single base learner (Freund and Schapire, 1996), (Dietterich, 2000). Ensembles can be constructed as parallel or serial collections of base learners. A parallel ensemble combines independently constructed base learners. Because different errors can cancel each other, an ensemble of such base learners can outperform any single one of its components (Hansen and Salamon, 1990), (Amit and Geman, 1997). Parallel ensembles are often applied to high-variance base learners (such as trees). (Valentini and Dietterich, 2003) showed that ensembles of low-bias support vector machines (SVMs) often outperformed a single, best-tuned, canonical SVM (Boser et al., 1992).

A well-known example of a parallel ensemble is a random forest (RF) (Breiman, 2001). It uses subsampling and to build a collection of trees and injects additional randomness through a random selection of variable candidates of each node of each tree. The forest can be considered a more sophisticated bagging method (Breiman, 1996). It is related to random subspace method of (Ho, 1998). A forest of random decision trees are grown on bagged samples with peroformance comparable to the best known classifiers. Given M predictors a RF can be summarized as follows: (1) Grow each tree on a bootstrap sample of the training set to maximum depth, (2) Select at random $m < M$ predictors at each node, and (3) Use the best split selected from the possible splits on these m variables. Note that for every tree grown in RF, about one-third of the cases are out-of-bag (out of the bootstrap sample). The out-of-bag (OOB) samples can serve as a test set for the tree grown on the non-OOB data.

In serial ensembles, every new learner is based on the prediction errors from previously built learners so that the weighted combination forms an accurate model. A serial ensemble results in an additive model built by a forward-stagewise algorithm and *Adaboost* introduced by (Freund and Schapire, 1996) is the best-known example.

Neither parallel nor serial ensembles alone are sufficient to generate an adequate best subset model that accounts for masking, and detects more subtle effects. A simple example by (Tuv, 2006) illustrated this. In some cases, weak but independent predictors are incorrectly promoted in the presence of strong, but related predictors. In other cases the weak predictors are not detected. An integrated solution is expected to provide advantages and several concepts described previously were integrated into a best subset selection algorithm by (Tuv et al., 2007). Only a brief summary is provided here. The best-subset algorithm contains the following steps:

1. Variable importance scores are computed from a parallel RF ensemble. Each tree uses a fixed depth of 3-6 levels. There are some modified calculations based on OOB sample that are described in more detail by (Tuv et al., 2007).
2. Noise variables are created through a random permutation of each column of the actual data. Because of this random permutation, the noise variables are known to not be associated with the target. The noise variables are used to set a threshold for statistically significant variable importance scores to select important (relevant) variables
3. Within decision trees, surrogate scores can be calculated from the association between the primary splitter at a node and other potential splitters. The details were originally provided by (**?**). These surrogate scores describe how closely an alternative splitter can mach the primary. This is turn provides a measure of masking between these variables. When such scores are combined from all nodes in a tree and all trees in an ensemble, a robust metric for variable masking can be obtained. A masking matrix is computed and noise variables are again used to determine significance thresholds. A set of short serial ensemble is used.
4. Masked variables are removed from the list of important variables
5. The target is adjusted for the currently identified important variables, and the algorithm is repeated. The adjustment calculates generalized residuals that apply to either regression of classification problems. Less important variables can be more clearly identified once the dominant contributors are eliminated. Trees-based models are not well-suited for additive models and the iteration substantially improves the performance in these cases.

4 ILLUSTRATIVE EXAMPLE

Because change-point detection is an unsupervised learning task, simulated data is used with known change points inserted. A data set to mimic a real manufacturing environment includes 10 sensors that each generate time series (with 100 time data points) from given distributions. Each time series could be represented as a trapezoid with added curvatures, an oscillation with random phase in the center, and Gaussian noise on the order of 10% of the signal. Curvatures and the center oscillation phase are sampled

from fixed uniform distributions. This set of time series provides the results for one batch and the objective is to detect changes in a series of batches. The dataset consists of 10000 batches and each 1000 samples there is a change induced by a shift in one of internal parameters used to generate time series by its standard deviation.

Such high-dimensional data can be analyzed directly, or a different representation can be used to extract features that might be of interest. For example, Fourier transforms, discrete wavelet transforms, and orthogonal polynomials are only a few of the methods to represent high-dimensional data. Without a priori information of features affected by a change point, the set of features is extracted from these methods is still often quite large.

Chebyshev polynomials are used here to represent this high-dimensional data. The representation is $y(x) = T_n(x)$ where $T_n(x)$ by definition is a polynomial solution of degree n of the equation

$$(1-x^2)\frac{d^2y}{dx^2} - x\frac{dy}{dx} + n^2 y = 0, \qquad (2)$$

where $|x| \leq 1$ and n is a non-negative integer. For $n = 0$ $T_0(x) = 1$. Chebyshev polynomials can also be calculated using one of useful properties: $T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x)$ and $T_n(x) = \cos(n \cdot \cos^{-1}(x))$.

A set of Chebyshev polynomials $\{T_n(x)\}_{n=0,1,\ldots}$ is orthogonal with respect to the weighting function $(1-x^2)^{-1/2}$:

$$\int_{-1}^{1} \frac{T_m(x)T_n(x)dx}{\sqrt{1-x^2}} = \begin{cases} \frac{1}{2}\pi\delta_{nm}, n>0, m>0 \\ \pi, n=0, m=0 \end{cases}, \qquad (3)$$

where δ_{mn} is the Kronecker delta.

Using the last property we can represent any piecewise continuous function $f(x)$ in the interval $-1 \leq x \leq 1$ as a linear combination of Chebyshev polynomials:

$$\sum_{0}^{\infty} C_n T_n(x) = \begin{cases} f(x), \text{where } f(x) \text{ is continuous} \\ \frac{f(x-0)+f(x+0)}{2} \text{ in discontinuity points} \end{cases} \qquad (4)$$

Here

$$C_n = \frac{A}{\pi}\int_{-1}^{1} \frac{f(x)T_n(x)dx}{\sqrt{1-x^2}}, \quad A = \begin{cases} 1, n = 0 \\ 2, n > 0 \end{cases}. \qquad (5)$$

For a function $\{f_i\}_{i=1,\ldots,P}$ defined on a discrete domain we calculate the coefficients of the Chebyshev decomposition using a straightforward formula:

$$C_n = \frac{A}{\pi}\sum_{i=1}^{T} \frac{f_i T_n(x_i)}{\sqrt{1-x_i^2}}, \qquad (6)$$

where $x_i = -1 + \frac{2}{P}(i - \frac{1}{2})$. Therefore, the coefficients $\{C_n\}$ become the features for the change-point detection. We use first 25 coefficients for each time series resulting in 250 features for each sample.

In the first phase of the analysis the feature selection algorithm simply uses a sequential batch index as the target. The polynomial coefficients provide the inputs. The feature selection module identifies the distribution change and a set of features responsible for the change.

In the second phase, moving T^2 statistics are calculated using only the selected features between n_1/n_2-samples prior/after the current data point, correspondingly, to detect the change points:

$$T^2 = \frac{n_1 n_2 (n_1+n_2-2)}{n_1+n_2} (\overline{\mathbf{y}}_1 - \overline{\mathbf{y}}_2)' \mathbf{W}^{-1} (\overline{\mathbf{y}}_1 - \overline{\mathbf{y}}_2), \qquad (7)$$

where

$$\mathbf{W} = \sum_{j=1}^{n_1} (\mathbf{y}_{1j} - \overline{\mathbf{y}}_1)(\mathbf{y}_{1j} - \overline{\mathbf{y}}_1)' + \sum_{j=1}^{n_2} (\mathbf{y}_{2j} - \overline{\mathbf{y}}_2)(\mathbf{y}_{2j} - \overline{\mathbf{y}}_2)'. \qquad (8)$$

In the second phase, moving T^2 statistics are calculated using only the selected features between n_1/n_2-samples prior/after the current data point, correspondingly, to detect the change points:

$$T^2 = \frac{n_1 n_2 (n_1+n_2-2)}{n_1+n_2} (\overline{\mathbf{y}}_1 - \overline{\mathbf{y}}_2)' \mathbf{W}^{-1} (\overline{\mathbf{y}}_1 - \overline{\mathbf{y}}_2), \qquad (9)$$

where

$$\mathbf{W} = \sum_{j=1}^{n_1} (\mathbf{y}_{1j} - \overline{\mathbf{y}}_1)(\mathbf{y}_{1j} - \overline{\mathbf{y}}_1)' + \sum_{j=1}^{n_2} (\mathbf{y}_{2j} - \overline{\mathbf{y}}_2)(\mathbf{y}_{2j} - \overline{\mathbf{y}}_2)'. \qquad (10)$$

We retrain a model with each 200 samples using all samples from the previous detected change point until the current sample. We do not make predictions on the first 100 samples in the beginning and after a change point. The results are shown in Figure 1. Here T^2 is shown with feature selection in the top figure and without feature selection in the bottom figure. Notice that changes are not detected before feature selection improves the sensitivity of the control chart. After feature selection the changes are apparent.

5 CONCLUSIONS

As sensors continue to flourish in numerous disciplines, high-dimensional data becomes more common. Furthermore, the ability to detect changes in

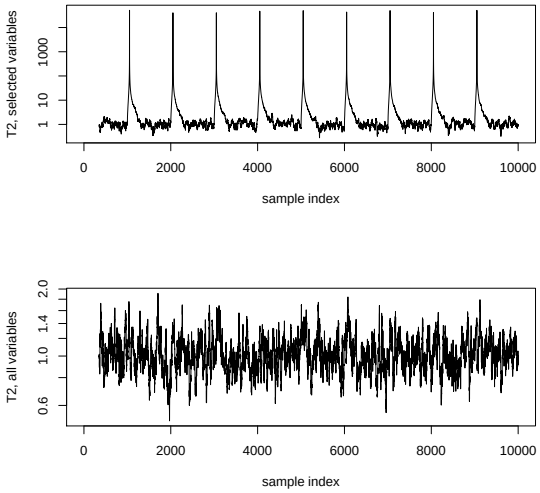


Figure 1: Time series plot of T^2 with (top) and without (bottom) feature selection.

a system or process over time remains an important need in many applications. The results here illustrate the success of a solution that integrates several important elements. The transform of the inherently unsupervised learning problem of change-point detection to one of supervised learning with a time index as the response, opens the analysis to a wide collection o tools. A sophisticated feature selection algorithm can then be applied to detect attributes that contribute to a change. In the lower-dimensional space of these attributes, the change point detection is a much simpler problem and a number of simpler tools can be applied. We uses a multivariate T^2 control chart, but other control charts, or methodologies can be considered after the important dimensional reduction. The illustrative example presents an simulation of an important practical case. One needs to summarize the information from multiple time series. Consequently, the dimensional space equals the number of series times the length of each series and the feature selection becomes critical, and the example illustrates an effective solution method for this problem.

ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 0355575.

REFERENCES

Amit, Y. and Geman, D. (1997). Shape quantization and recognition with randomized trees. *Neural Computation*, 9(7):1545–1588.

Belisle, P., Joseph, L., Macgibbon, B., Wolfson, D. B., and Berger, R. D. (1998). Change-point analysis of neuron spike train data. *Biometrics*, 54:113–123.

Boser, B., Guyon, I., and Vapnik, V. (1992). A training algorithm for optimal margin classifiers. In Haussler, D., editor, *5th Annual ACM Workshop on COLT, Pittsburgh, PA*, pages 144–152. ACM Press.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24(2):123–140.

Breiman, L. (2001). Statistical modeling: The two cultures. *Statistical Science*, 16(3):199–231.

Dietterich, T. G. (2000). An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, 40(2):139–157.

Freund, Y. and Schapire, R. E. (1996). Experiments with a new boosting algorithm. In *the 13th International Conference on Machine Learning*, pages 148–156. Morgan Kaufman.

Guyon, I. and Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182.

Hansen, L. K. and Salamon, P. (1990). Neural network ensembles. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 12(10):993–1001.

Ho, T. K. (1998). The random subspace method for constructing decision forests. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(8):832–844.

Hotelling, H. (1947). Multivariate quality control-illustrated by the air testing of sample bombsights. *Techniques of Statistical Analysis*, pages 111–184.

Li, F., Runger, G. C., and Tuv, E. (2006). Supervised learning for change-point detection. *IIE Transactions*, 44(14-15):2853–2868.

Liu, H. and Yu, L. (2005). Toward integrating feature selection algorithms for classification and clustering. *IEEE Trans. Knowledge and Data Eng.*, 17(4):491–502.

Pievatolo, A. and Rotondi, R. (2000). Analysing the interevent time distribution to identify seismicity phases: a bayesian nonparametric approach to the multiple change-points problem. *Applied Statistics*, 49(4):543–562.

Tuv, E. (2006). Ensemble learning and feature selection. In Guyon, I., Gunn, S., Nikravesh, M., and Zadeh, L., editors, *Feature Extraction, Foundations and Applications*. Springer.

Tuv, E., Borisov, A., Runger, G., and Torkkola, K. (2007). Best subset feature selection with ensembles, artificial variables, and redundancy elimination. *Journal of Machine Learning Research*. submitted.

Valentini, G. and Dietterich, T. (2003). Low bias bagged support vector machines. In *ICML 2003*, pages 752–759.

INVERSION OF A SEMI-PHYSICAL ODE MODEL

Laurent Bourgois, Gilles Roussel and Mohammed Benjelloun
Laboratoire d'Analyse des Systèmes du Littoral (EA 2600)
Université du Littoral - Côte d'Opale
50 rue Ferdinand Buisson, B.P. 699, 62228, Calais Cedex, France
firstname.name@lasl.univ-littoral.fr

Keywords: Semi-physical modeling, gray-box, inverse dynamic model, neural network, model fusion.

Abstract: This study proposes to examine the design methodology and the performances of an inverse dynamic model by fusion of statistical training and deterministic modeling. We carry out an inverse semi-physical model using a recurrent neural network and illustrate it on a didactic example. This technique leads to the realization of a neural network inverse problem solver (NNIPS). In the first step, the network is designed by a discrete reverse-time state form of the direct model. The performances in terms of generalization, regularization and training effort are highlighted in comparison with the number of weights needed to estimate the neural network. Finally, some tests are carried out on a simple second order model, but we suggest the form of a dynamic system characterized by an ordinary differential equation (ODE) of an unspecified *r* order.

1 INTRODUCTION

Generally, inverse problems are solved by the inversion of the direct knowledge-based model. A knowledge-based model describes system behavior using the physical, biological, chemical or economic relationships formulated by the expert. The "*success*" of the data inversion, i.e. the restitution of a nearest solution in the sense of some ℓ_2 or ℓ_∞ norm distance from the exact sources of the real system, depends on the precision of the model, on the noise associated with the observations and on the method.

Whereas the noise is inherent in the hardware and conditions of measurement and thus represents a constraint of the problem, on the other hand, the two controls an engineer possesses to improve quality of the estimated solution are the model and the method. The approach we propose thus relates to these two aspects. It aims at overcoming several difficulties related to the definition of the model and its adjustments, and to the search for a stable solution of the sought inputs of the system.

2 SEMI-PHYSICAL MODELING

Obtaining a robust knowledge-based model within the meaning of exhaustiveness compared to the variations of context (one can also say generic), is often tricky to express for several reasons. One firstly needs a perfect expertise of the field to enumerate all the physical laws brought into play, all the influential variables on the system and an excellent command of the subject to make an exhaustive spatial and temporal description of it. Even if the preceding stage is completed, it is not rare that some parameters can not be measured or known with precision. It is then advisable to estimate these parameters starting from the observable data of the system under operation. Once the physical model has been fixed, it is endowed with good generics.

A black-box model is a behavior model and depends on the choice of a mathematical *a priori* form in which an engineer has a great confidence on its adaptability with the real behavior of the system. In the black-box approach, the model precision is thus dependent on the adopted mathematical form, on the approximations carried out on the supposed system order (linear case), on the assumptions of nonlinearity, and on the quantity and the quality of data to make the identification of the model. Many standard forms of process (ARMA, ARMAX, NARMAX) (Ljung, 1999) are able to carry out a black-box modeling. Other techniques containing neural networks have the characteristic not to specify a mathematical form but rather a neural structure adapted to the nature of the system (static, dynamic, linear, nonlinear, exogenic

input, assumptions on the noise). Neural networks are known for possessing a great adaptability, with the properties of universal approximator (Hornik et al., 1989), (Sontag, 1996) if the available training examples starting from validated observations are in a significant number. Nevertheless, black-box models are less parsimonious than knowledge-based models since the mathematical functions taking part in the latter are exact functions (not leaving residues of the output error, without noise). The second disadvantage of neural black-box models is the least generic behavior with respect to the new examples which do not form part of the base of training set. However some techniques exist to improve the generalizing character of a neural network, like the regularization or the abort (early stopping) of the training when the error of validation increases.

Between the two types of models previously exposed, (Oussar and Dreyfus, 2001) introduce the semi-physical or gray-box model. This type of model fulfills at the same time the requirements of precision, generics, parsimony of the knowledge-based models, and also possesses the faculty of training and adaptability. Close approaches were proposed by (Cherkassky et al., 2006). These approaches often consist in doing the emulation of physically-based process models starting from training of neural networks with simulated data (Krasnopolsky and Fox-Rabinovitz, 2006). If the knowledge model is difficult to put in equation because of its complexity, the idea will be to structure a looped neural network (case of dynamic complex systems) using knowledge on the fundamental laws which govern the system. Then, we add degrees of freedom (neurons) to the network to adapt it to the ignored parts of the system. The recall phase (production run) then makes it possible to carry out the predicted outputs in real time.

3 INVERSE NEURAL MODEL

3.1 Principle

The inversion of a physical model generally consists in estimating information on the nonmeasurable parameters or inputs starting from the measurable observations and *a priori* information on the system. We propose here to use the training of an inverse model using a neural network. Some ideas for forward and inverse model learning in physical remote measurement applications are proposed by (Krasnopolsky and Schillerb, 2003). It consists in estimating parameters of the network so that the outputs correspond to the inputs (or the parameters) desired for training set of examples (figure 1). In recall phase, the network estimates the amplitudes of the parameters or the sequence of the input vector for the measured observations (figure 2), by supposing here that the real model does not evolve any more after the last training. Here the model is structured by the inverse model starting from the direct deterministic model.

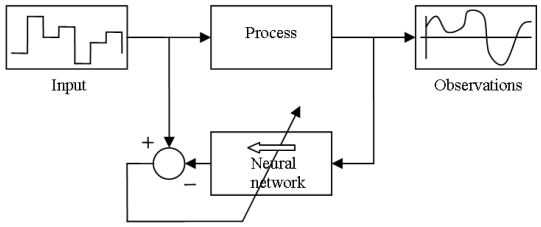


Figure 1: Training phase of the inverse neural model.

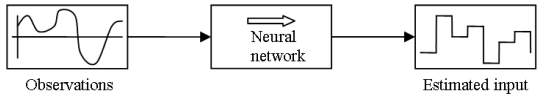


Figure 2: Recall phase of the inverse neural model.

3.2 Regularization and Inverse Neural Model

Inverse problems are often ill posed within the meaning of Hadamard (Groetsch, 1993). They can present:

1. An absence of solution;
2. Multiple solutions;
3. An unstable solution.

To transform ill posed problems into well conditioned problems, it is necessary to add *a priori* knowledge on the system to be reversed. There are several approaches which differ by the type of *a priori* knowledge introduced (Thikhonov and Arsenin, 1977), (Idier, 2001), (Mohammad-Djafari et al., 2002).

However, can we pose the problem of the regularization in the case of the NNIPS ? In fact, it is clear that the neural network provides a solution to the presentation of an input example. Even if this example is unknown, the network answers in a deterministic way by a solution, which could be false. From its property of classifier and autoassociativity, it will provide in best case, the most similar solution to the class including the test examples. That thus answers difficulties 1 and 2 of the inverse problems, even if the suggested solution can prove to be false. In addition, we saw above that regularization during training phase, improves generalization with respect to the examples.

It avoids the problem of overtraining which precisely results in an instability of the solutions in the vicinity of a point. It is remarkable that the early stopping method should have an interesting effect on generalization and constitute a particular form of the regularization. In other words, we have found, with the neural networks the techniques usually exploited for the analytical or numerical inverse problems regularization so as to answer risk 3. This confirms our opinion to use the neural networks like an inverse model.

4 RECURRENT NEURAL SYSTEMS MODELING

4.1 General Case

In particular, we have been interested in dynamic systems represented by recurrent equations or finite differences. We have chosen to represent the models by a state space representation because of systems modeling convenience and for the more parsimonious character compared to the input-outputs transfer type models. We have thus made the assumption that the system can be represented by the following state equations:

$$\begin{cases} x(n+1) &= \varphi[x(n),u(n)] \\ y(n) &= \psi[x(n)]+b(n) \end{cases} \quad (1)$$

φ is the vector transition function, ψ is the output vector function and $b(n)$ is the output noise to instant n. Under this assumption of output noise, neural modeling takes the following canonical form (Dreyfus et al., 2004):

$$\begin{cases} x(n+1) &= \varphi_{RN}[x(n),u(n)] \\ y(n) &= \psi_{RN}[x(n)]+b(n) \end{cases} \quad (2)$$

The observation noise appearing only in the observation equation does not have any influence on the dynamic of the model. In this case, the ideal model is the looped model, represented on figure 3.

4.2 Dynamic Semi-Physical Neural Modeling

The semi-physical model design requires that one should have a knowledge-based model, usually represented in the form of an algebraic equation whole, differential, with partial derivative, sometimes nonlinear coupled. We have examined the modeling of a system represented by an ordinary differential equation. To expose the principle, we have again taken

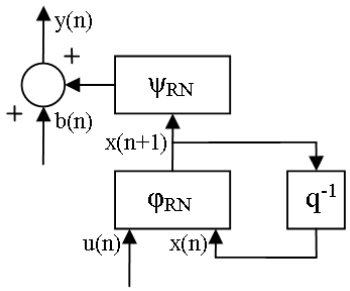


Figure 3: Ideal direct neural model with output noise assumption. The q^{-1} operator stands for one T sample time delay.

the essential phases of semi-physical neural modeling more largely exposed in (Oussar and Dreyfus, 2001), (Dreyfus et al., 2004). We have also supposed that the starting model can be expressed by the continuous state relations:

$$\begin{cases} \frac{dx}{dt} &= f[x(t),u(t)] \\ y(t) &= g[x(t)] \end{cases} \quad (3)$$

Where x is the vector of state variables, y is the output vector, u is the command inputs vector, and where f and g are vector functions. The functions f and g can however be partially known or relatively vague. In a semi-physical neural model, the functions which are not precisely known are fulfilled by the neural network, after the preliminary training of the latter from experimental data. The accurately known functions are maintained in their analytical form, but one can also adopt a neural representation whose activation function is known and does not use of adjustable parameters. The design of a semi-physical model generally includes four stages:

- Obtaining the discrete knowledge-based model;
- Designing the network in the canonical form (2) by adding degrees of freedom;
- Initializing from a knowledge-based model simulator;
- Training from the experimental data.

We have applied these steps to an inverse semi-physical model by adding a stage of inversion of the discrete model before training.

4.3 An Academic Example: the Direct Second Order Ode Model

We have studied the deconvolution problem for linear models governed by an ordinary differential equation in order to test the method. However, this work has been only one first step with more general inverse

problem, which aims, starting from some observations of the system at carrying out the training of the inverse model for then being able to estimate the inputs and the observable states (within the observability sense of the states) of the system. Let us suppose a system represented by the differential equation:

$$\frac{d^2y}{dt^2}+2\xi\omega_n\frac{dy}{dt}+\omega_n^2 y = c_1 u(t) \tag{4}$$

This second order differential equation could be the representation of a mechanical system (mass, spring, shock absorber) or of an electric type (RLC filter) excited by a time depending input $u(t)$. This physical model where the kinematic parameters of damping ξ, natural pulsation ω_n, and static gain c_1 are not *a priori* known, can be represented by the model of following state:

$$\begin{cases} \dfrac{dx(t)}{dt} = \begin{bmatrix} 0 & 1 \\ -\omega_n^2 & 1-2\xi\omega_n \end{bmatrix} x(t)+\begin{bmatrix} 0 \\ c_1 \end{bmatrix} u(t) \\ y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} x(t) \end{cases} \tag{5}$$

The first stage is supplemented by the discretization. By supposing that we have collected the data with T sampling period, we have proceeded to the discretization by choosing the explicit Euler method. For the system (5) to which one has added an observation noise $b(n)$, the discrete equation of state is obtained:

$$\begin{cases} x(n+1) = Fx(n)+Gu(n) \\ y(n) = Hx(n)+b(n) \end{cases} \tag{6}$$

With:

$$\begin{cases} F = \begin{bmatrix} 1 & T \\ -\omega_n^2 T & 1-2\xi\omega_n T \end{bmatrix} \\ G^T = \begin{bmatrix} 0 & T \end{bmatrix} \\ H = \begin{bmatrix} c_1 & 0 \end{bmatrix} \end{cases} \tag{7}$$

The model in the form of looped neural network of the nondisturbed canonical system (6), is represented on figure 4.

The transfer functions represented on figure 4 are purely linear, being the ideal neural model. No new degrees of freedom are added to the direct model, this one being only intermediate representation, nonessential to the study. It simply illustrates with an example, the general form of the figure 3.

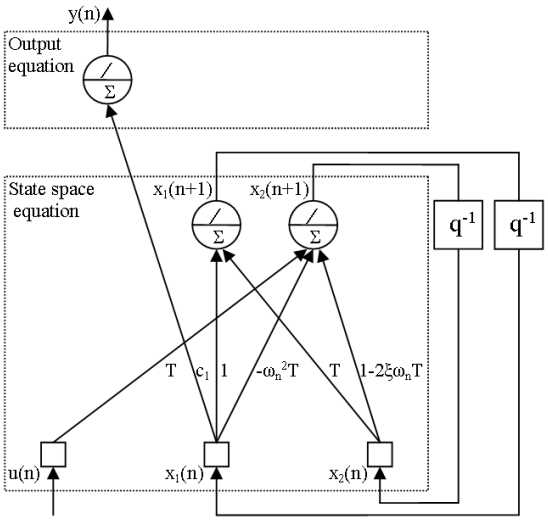


Figure 4: Direct second order neural model.

5 INVERSE SEMI-PHYSICAL NEURAL MODEL

5.1 Case of Second Order Model

From the preceding model we have expressed the output $u(n)$ according to the input $y(n)$. The particular shape of Kronecker of matrices G and H in the relation (6), has enabled us to isolate $x_1(n)$, $x_2(n)$ and $u(n)$:

$$\begin{cases} x_1(n) = \dfrac{y(n)-b(n)}{c_1} \\ x_2(n) = \dfrac{x_1(n+1)}{T}-\dfrac{y(n)-b(n)}{c_1 T} \\ u(n) = \alpha x_1(n+1)+\beta x_2(n+1)+\gamma[y(n)-b(n)] \end{cases} \tag{8}$$

With:

$$\begin{cases} \alpha = \dfrac{2\xi\omega_n T-1}{T^2} \\ \beta = \dfrac{1}{T} \\ \gamma = \dfrac{(\omega_n T)^2+(1-2\xi\omega_n T)}{c_1 T^2} \end{cases} \tag{9}$$

And finally, one has obtained the matrix form:

$$\begin{cases} x(n) = \begin{bmatrix} 0 & 0 \\ \frac{1}{T} & 0 \end{bmatrix} x(n+1)+\begin{bmatrix} \frac{1}{c_1} \\ -\frac{1}{c_1 T} \end{bmatrix} [y(n)-b(n)] \\ u(n) = \begin{bmatrix} \alpha & \beta \end{bmatrix} x(n+1)+\gamma[y(n)-b(n)] \end{cases} \tag{10}$$

Of course, this noncausal equation is realistic only if we calculate the recurrence by knowing the state at

the moment $n+1$ to determine the state at the moment n. This is more natural to the inverse problem where we seek to reconstitute the input sequence at the origin of the generated observations. The equation now reveals the output noise as a correlated state noise $b(n)$ and as a noise on the input $u(n)$ with a rate of amplification equivalent to the real γ. The ideal looped neural network representation of the model (10) is given on figure 5. We preserve the delays q^{-1} between the states $x(n+1)$ and $x(n)$ because of the presentation of the output $y(n)$ and the calculation of $u(n)$ in reverse-time. In addition, this model remains stable for any T, the eigenvalues of the reverse-time state matrix being null for this example.

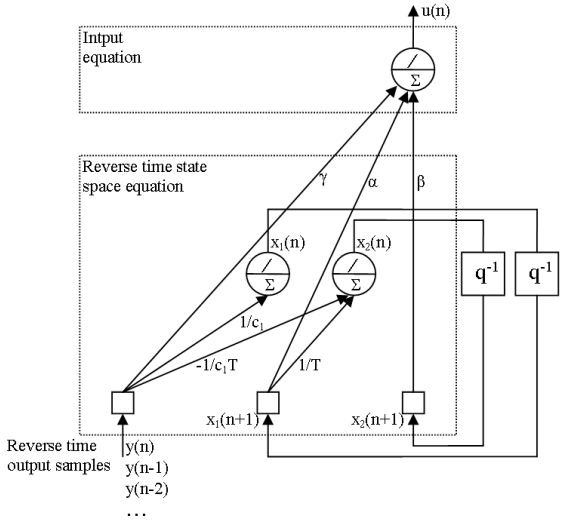


Figure 5: Inverse second order neural model.

If the sampling period is generally known, coefficients α, β and γ of the physical model can be imprecise, or completely unknown. The degrees of freedom that can be added to the network can relate to these parameters, themselves resulting from a combination of physical parameters. It is then optionally advisable to supplement this network by adding additional neurons on some internal links where parameters must be estimated.

5.2 General Case for r Order Ode without Derivative Input

The general case of the ODE mono input, mono output, continuous, without derivative from the input is expressed as follows:

$$a_r \frac{d^r y}{dt^r} + a_{r-1} \frac{d^{r-1} y}{dt^{r-1}} + \cdots + a_1 \frac{dy}{dt} + a_0 y = c_1 u(t) \quad (11)$$

In time discretization by sample interval of a low width T by the Euler's finite differences method, the shape of the direct equation matrices F, G and H are then:

$$\begin{cases} F = \begin{bmatrix} 1 & T & 0 & & & 0 \\ 0 & 1 & T & & & \\ & & \ddots & & & \\ & & & 1 & T & 0 \\ 0 & & & 0 & 1 & T \\ -\frac{a_0 T}{a_r} & & \cdots & & & -\frac{a_{r-1} T}{a_r} + 1 \end{bmatrix} \\ G^T = \begin{bmatrix} 0 & \cdots & 0 & T \end{bmatrix} \\ H = \begin{bmatrix} c_1 & 0 & \cdots & 0 \end{bmatrix} \end{cases} \quad (12)$$

A new system is obtained with the inverse model expressed in reverse-time:

$$\begin{cases} x(n) = F_I x(n+1) + G_I [y(n) - b(n)] \\ u(n) = H_I x(n+1) + I_I [y(n) - b(n)] \end{cases} \quad (13)$$

Where the matrices of the inverse state equation F_I, G_I, H_I and I_I are all dependent on T. The retrograde lower triangular state matrix, of size $dim(F_I) = r \times r$ and of rank $(r-1)$, takes the form (14).

$$F_I = \begin{bmatrix} 0 & & & & 0 \\ \frac{1}{T} & 0 & & & \\ -\frac{1}{T^2} & \frac{1}{T} & \ddots & & \\ & & & 0 & \\ -\left(-\frac{1}{T}\right)^{r-1} & -\left(-\frac{1}{T}\right)^{r-2} & \cdots & \frac{1}{T} & 0 \end{bmatrix} \quad (14)$$

The output application matrix G_I, of dimension $dim(G_I) = r \times 1$ becomes (15).

$$G_I^T = \begin{bmatrix} \frac{1}{c_1} & -\frac{1}{c_1 T} & \frac{1}{c_1 T^2} & \cdots & \frac{1}{c_1}\left(-\frac{1}{T}\right)^{r-1} \end{bmatrix} \quad (15)$$

The input matrix of dimension $dim(H_I) = 1 \times r$ is worth (16).

$$H_I = \begin{bmatrix} 0 & \cdots & 0 & \frac{1}{T} \end{bmatrix} + \begin{bmatrix} \frac{a_0}{a_r} & \cdots & \frac{a_{r-2}}{a_r} & -\frac{1}{T}\left(1 - \frac{a_{r-1} T}{a_r}\right) \end{bmatrix} F_I \quad (16)$$

The direct application matrix I_I of the output to the input, of dimension $dim(I_I) = 1 \times 1$ is given by (17).

$$I_I = \left[\frac{a_0}{a_r} \quad \cdots \quad \frac{a_{r-2}}{a_r} \quad -\frac{1}{T}\left(1-\frac{a_{r-1}T}{a_r}\right) \right] G_I \tag{17}$$

If it is considered that the representation of the inverse model (1) takes the form (18), the general neural representation of the inverse model takes the form shown on figure 6.

$$\begin{cases} x(n) = \varphi^I_{RN}[x(n+1), y(n)] \\ u(n) = \psi^I_{RN}[x(n+1), y(n)] \end{cases} \tag{18}$$

In Which $\varphi^I_{RN}(F_I, G_I)$ is related to transition to reverse-time and $\Psi^I_{RN}(H_I, I_I)$ represents the restoring function of the input. On the numerical level, the eigenvalues of the matrix F_I in ODE case being all null, thus there is no risk of instability.

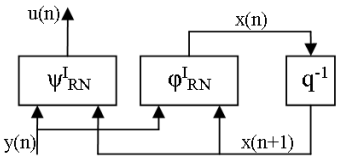


Figure 6: Neural representation of the reverse state model.

6 RESULTS

The goal of this section is to check the assumptions of awaited quality concerning the two NNIPS (black-box and gray-box models) in terms:

- of robustness with respect to an unknown input from the base of training compared to the inverse state model;
- of robustness with respect to the noise on the output (i.e. the regularizing effect) compared to the inverse state model;
- of gain in effort of training.

The black-box neural model is an Elman network with two linear neurons on its hidden layer, and with one neuron on its output layer. This neural network is fully connected, and the weights and biases are initialized with the Nguyen-Widrow layer initialization method (Nguyen and Widrow, 1990). The semi-physical model design is carried out from the preceding black-box model which has been modified to obtain the neural representation of the figure 5. For that, the coefficients depending on the parameters T, c_1, α, β and γ are left free. Only three coefficients have been forced to be null to delete corresponding connections. We have also connected the inputs layer to the output layer, but we have not added any additional neuron. These two models are subjected to a training with (pseudo) experimental disturbed data. For the numerical tests, we have adopted the parameters according to $\omega_n = 5\ rad.s^{-1}$, $\xi = 0.4$, $T = 0.05\ s$ and $c_1 = 1$. It is noticed that this choice of parameters ensures, for the matrix F of the system (6), a spectral radius lower than 1, and consequently the stability of the direct model.

To construct the sets of training, we have generated a N samples input random sequence to simulate the direct knowledge-based model. This signal is a stochastic staircase function, resulting from the product of an amplitude level A_e by a Gaussian law of average μ_e and variance σ_e^2, of which the period of change T_e of each state is adjustable. T_e influences the input signal dynamics, and thus the spectrum of the system excitation random signal. For all the tests, we have fixed $A_e = 1$, $\mu_e = 0$, $\sigma_e^2 = 1$, $T_e = 60T$, $T_b = 3T$ and $\mu_b = 0$. This input signal provides a disturbed synthetic output signal. The variance σ_b^2, the average μ_b, as well as the period of change of state T_b characterize the dynamics of the noise.

Weights Initial Value: The stage of coefficients initialization being deterministic for the quality of the results in the black-box model case, we have chosen to reproduce hundred times each following experiments. Indeed, some initial values can sometimes generate mean squared error (MSE) toward infinite value. These results will then be excluded before carrying out performances and average training efforts calculation.

6.1 Test On Modeling Errors and Regularizing Effect

We have measured the generalization and regularization contribution of the inverse neural model compared to the inverse state model. For that, we have compared the mean square errors of the inverse state model with those obtained in phase of recall of the two inverse neural form models.

6.1.1 Training and Test Signals and Comparison of Restorations

We have tested five training sequences length $N = 300$ samples. The variances which characterize the dynamics of the noise in the pseudo experimental signals σ_b^2 are worth 0, 0.03, 0.09, 0.25, and 1. They generate for the process output signal several values of signal to noise ratio (SNR) from around 20 dB to infinity. Then we have compared the MSE of deconvolution in recall phase with new disturbed random

signals with the same variances and applied to the inverse black-box and semi-physical models.

6.1.2 Numerical Results

Let us underline the fact that only seventy-six experiments have been retained for calculation of averages. The black-box model have not provided (due to a bad initialization of the coefficients) suitable restoration in 24% of the cases. The figure 7 gathers the results of MSE for the three inverse models. The figure 8 illustrates the signals obtained for the output signals deconvolutions with a SNR of 33 dB.

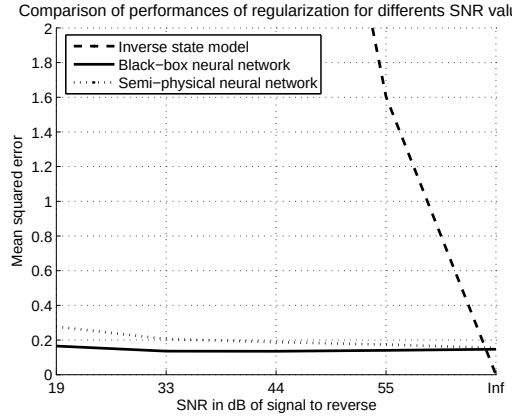


Figure 7: Impact of neural models on the regularization: evolution of the three models MSE according to the SNR.

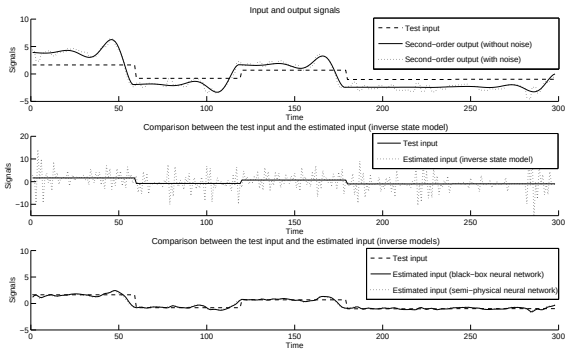


Figure 8: a) Test signals (SNR of 33 dB), b) Deconvolution by the reverse state model, c) Deconvolution by the inverse neural models.

Noiseless case: In absence of noise in training and test signals, black-box and semi-physical models provide similar mean performances (MSE of approximately 0.18) and in addition relatively near to the inverse state model.

Noisy case: When the noise grows in the training and test signals, the two neural models are much less sensitive to the noise than the inverse state model (figure 7). The regularizing effect is real. The semi-physical model has good performances but, the constraint imposed by the structure of the network and the more reduced number of connections (synapses), decreases the robust effect to the noise (loss of the neural network associative properties) and slightly places this model in lower part of the black-box model. It is thus noted that performances in term of regularization are much better than for the inverse state model, but a little worse than for the black-box model (MSE increases more quickly). The performances seem to be a compromise between the knowledge-based and the black-box model. Let us note that this difference grows with a higher order model and also increases the number of neurons.

6.2 Test On Learning Effort

For this test, we have compared the product of the MSE by the number of epochs, i.e. the final error amplified by the iteration count of the training phase. Learning stops if the iteration count exceeds 250 or if MSE is lower than $0,03$. We have made a distinction between errors at the end of the training (figure 9) and errors on the test as a whole (figure 10).

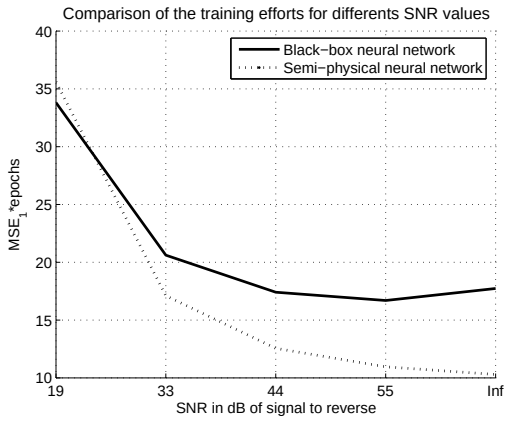


Figure 9: Curves of learning effort on the training set.

As learning effort depends on the number of neurons, one compares the black-box and gray-box networks with an equal number of neurons. In the first, all the weights of connections are unknown. In the second, one considers all the weights with the exception of the three coefficients corresponding to non-existent connections and being null. On figure 9, we note that the gray-box model is more effective when the noise is weak. Physical knowledge supports the convergence of the weights so that the behavior approaches the data. This seems to be checked until SNR of about 20 dB as in our example. Beyond

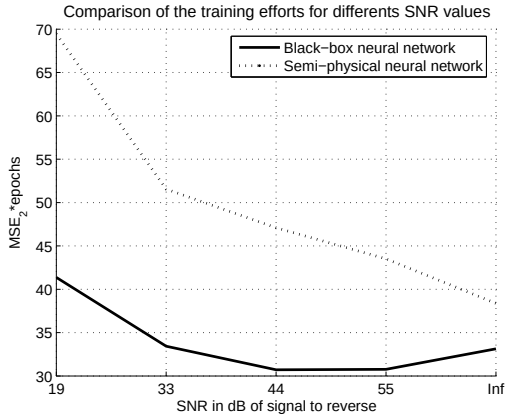


Figure 10: Curves of learning effort on the test set.

that, it is the black-box which is slightly more effective. On figure 10, the tendencies are the same for the black-box model. The gray-box model is penalized by the constraint imposed by the neural framework and the more reduced number of connections. Within the framework of a traditional second order ODE, the semi-physical model seems to require a greater effort with the test set than the black-box model. Indeed, within the framework of this example, the number of coefficients remaining free for the gray-box model being relatively low, it leads to a loss of the generalization capacity compared to the black-box model. However, this supremacy is quite relative since only 76% of the tests carried out have been conclusive for the black-box model and 100% for the gray-box model.

7 CONCLUSION

We have examined the performances of an inverse dynamic model by fusion of statistical learning and deterministic modeling. For the study, our choices have gone toward the design of an inverse semi-physical model using a looped neural network. We have compared the latter with an inverse fully connected neural network. Experimental results on a second order system have shown that the inverse gray-box neural model is more parsimonious and presents better performances in term of learning effort than the inverse black-box neural model, because of knowledge induced by the deterministic model. The performances in term of inverse modeling precisions are visible since the input restoration errors are weak. The neural training plays the part of statistical regressor and of regularization operator. Finally, a higher order model increases the number of neurons and then improves the robust effect to the noise of the gay-box model.

REFERENCES

Cherkassky, V., Krasnopolsky, V. M., Solomatine, D., and Valdes, J. (2006). Computational intelligence in earth sciences and environmental applications: Issues and challenges. *Neural Networks*, 19, issue 2:113–121.

Dreyfus, G., Martinez, J. M., Samuelides, M., Gordon, M. B., Badran, F., Thiria, S., and Hérault, L. (2004). *Réseaux de Neurones: Méthodologies et Applications*. Eyrolles, 2ème édition, Paris.

Groetsch, C. W. (1993). *Inverse Problems in the Mathematical Sciences*. Vieweg Sohn, Wiesbaden.

Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2:359–366.

Idier, J. (2001). *Approche Bayesienne pour les Problèmes Inverses*. Traité IC2, Série Traitement du Signal et de l'Image, Hermès, Paris.

Krasnopolsky, V. M. and Fox-Rabinovitz, S. F. (2006). Complex hybrid models combining deterministic and machine learning components for numerical climate modeling and weather prediction. *Neural Networks*, 19:122–134.

Krasnopolsky, V. M. and Schillerb, H. (2003). Some neural network applications in environmental sciences. part i: Forward and inverse problems in geophysical remote measurements. *Neural Networks*, 16:321–334.

Ljung, L. (1999). *System Identification, Theory for the User*. Prentice Hall, N. J.

Mohammad-Djafari, A., Giovannelli, J. F., Demoment, G., and Idier, J. (2002). Regularization, maximum entropy and probabilistic methods in mass spectrometry data processing problems. *Int. Journal of Mass Spectrometry*, 215, issue 1:175–193.

Nguyen, D. and Widrow, B. (1990). Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weigths. *Proceedings of the International Joint Conference on Neural Networks*, 3:21–26.

Oussar, Y. and Dreyfus, G. (2001). How to be a gray box : Dynamic semi-physical modeling. *Neurocomputing*, 14:1161–1172.

Sontag, E. D. (1996). Recurrent neural networks : Some systems-theoretic aspects. Technical Report NB, Dept of mathematics, Rutgers University, U.S.A.

Thikhonov, A. N. and Arsenin, V. Y. (1977). *Solutions of ill Posed Problems*. John Wiley, New York.

NONLINEAR FUZZY SELFTUNING PID CONTROL TECHNOLOGY AND ITS APPLICATIONS IN AUTOMATED PROGRAMMING ROBOTICS

Ganwen Zeng
Data I/O Corporation, 6464 185th Ave NE, Readmond, WA 98052, USA
zengg@dataio.com

Qianglong Zeng
Dartmouth University, Hanover, NH, 03755, USA
davidqlz@gmail.com

Keywords: Fuzzy control, Fuzzy sets, Fuzzy self-tuning control, Fuzzy PID control, Robotics, Distributed control, Programming systems, Control algorithm, Control system, DSP, ADC/DAC, Servo motor control, FPGA, Microcontrollers, PowerPC, ARM9, COM, CAN, ETHERNET, Multi-core architectures, Embedded control system, Bios, RTOS, kernel, OOP, Nested ISR, Fast IO, Firmware, Embedded software, Control applications, Programming technology, ONFI (Open NAND Flash Interface), Flash devices, Flash cards, Flash disks, USB2.0 host/Flash Media Controller for SmartMedia (SM), xD, CF, MS, SD and MultimediaCard (MMC).

Abstract: The paper presents an advanced Fuzzy self-tuning PID controller theory and it implement its applications on Data I/O's automated robotic programming systems. Considerable programming technology shift occurred in recent device programmer industry; programming density have been constantly fast growing from low-volume to high-volume programming for all kinds of non-volatile flash memory devices such as NOR flash, NAND flash, and MMC cards, SD flash cards, serial flash device, serial flash cards, flash-based microcontrollers and flash disks as high performance M-systems DiskOnChip. Device programming mode is more demanding an automatic programming than manual operation mode. It drives the creation and implementation of a high-performance automated programming robotic systems. This paper shows how this proposed advanced Fuzzy self-tuning PID controllers work on these automated programming robotic automation systems.

1 INTRODUCTION

Automated programming systems available today are able to fully automate device programming and to fully integrate programming testing, how to obtain a high control performance and good control system stability in these automated robotic system is one key of the success in long-term device operations in the programmer systems. It improves the productivity, quality and flexibility of a semiconductor production process.

High performance motor motion control precision and high level of integration is continuously increasing, and the clear trend is towards completely integrated intelligent programming system. This paper describes an embedded intelligent programming automation system. The robotic automated programmer system is shown in the Figure 1.

The robotic automated programmer system is implement in multiple microcontrollers, DSP and embedded processors, for an complex control motion and control task control, a multi-core architecture is used for a high-performance motion control and optimal marshalling control of multiple control tasks. The multi-core can easily assist a control task marshalling that implements a task-on-task control communication. The control block diagram of the robotic automated programmer system is shown in the Figure 2.

Figure 1: Data I/O FLX500.

The robotic automated programmer system is implement in multiple microcontrollers, DSP and embedded processors, for an complex control motion and control task control, a multi-core architecture is used for a high-performance motion control and optimal marshalling control of multiple control tasks. The multi-core can easily assist a control task marshalling that implements a task-on-task control communication. The control block diagram of the robotic automated programmer system is shown in the Figure 2.

It is composed of one host commander and two main motion control subsystems DPCS (Device Positioning Control System) and DIOC (Device Input Output Control System). The DPCS is composed by 5 control units of robotic task space configuration, X-gantry motion robotic control, y-gantry motion robotic control, head dynamic motion control, head rotation robotic control, device pick-and-place robotic control. The DIOC is composed by two control units of device feeding tape control and device transportation belt motion control. Synchronous communications can rely on the bus Ethernet and TCP/IP protocol in a multi-core architecture.

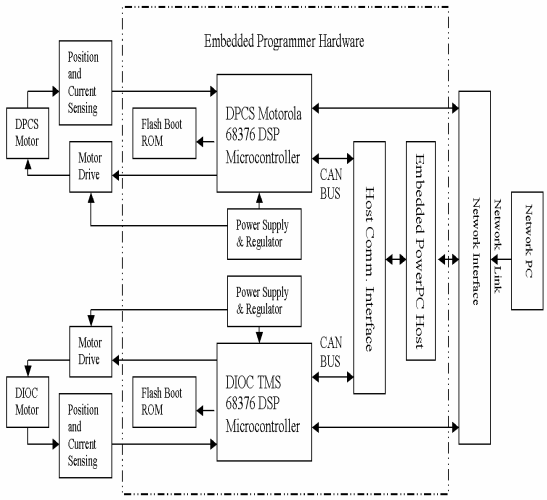


Figure 2: Control Block Diagram of Data I/O FLX500.

Asynchronous communications between the host commander and the motor control systems for the Nonlinear and Linear Brushless Servo Motors, AC Speed Motors with fractional HP, and Piezoelectric motors can be based on the TouCAN bus or Motorola 68376 Com Ports to guarantee the space loop closure for the main axes of the robotic control system. Based on the CAN architectures, a disturbed intelligent control structures is proposed in this multiple robotic axis configurations in the programmer automation system. This means the use of single-axis intelligent DSP motion controllers for both DPCS and DIOC which can handle local robotic axis control function independently from the multi-core processor host. Robust real-time OS kernel codes are used into the controllers to implement optimal interrupt service routines, fast IO, multi-threading, PWM generating units, current and motor torque control, speed/position control, and fuzzy control self-tuning PID control algorithms and integrated robotic motion solutions. Simulation has been done in LabVIEW 7.0 Professional Development System (PDS) and the MathWorks Matlab and Simulink. The real-time performances are shown in the conclusions.

2 CONTROL SYSTEM DESIGN AND RESEARCH

The control system design is shown in the Figure 3.

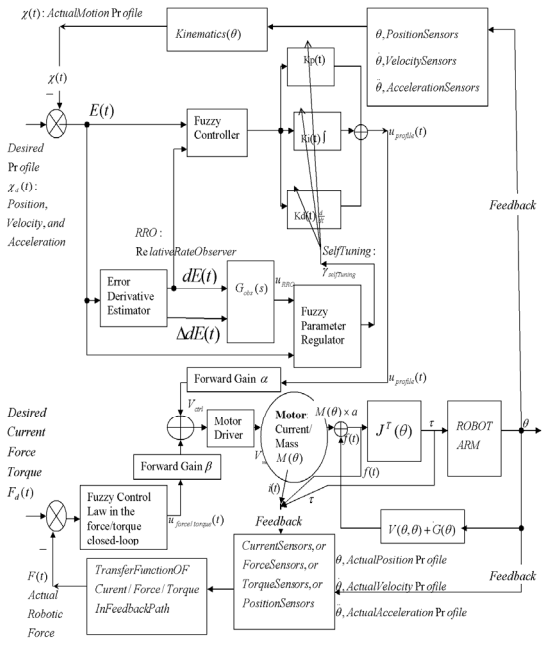


Figure 3: The control system design of nonlinear fuzzy self-tuning PID robotic controller.

The robotic control object to obtain a steady torque and high performance motion profiles, the robotic control system transfer function $G_{robot}(\theta)$ is:

$$G_{robot}(s) = \frac{\Theta(s)}{\tau(s)} = f_r(\Theta(t), \dot{\theta}(t), \ddot{\theta}(t), \tau(t))$$

And

$$f(t) = M_\chi(\theta)\ \ddot{\chi}(\theta(t)) + V_\chi(\theta, \dot{\vartheta}) + G_\chi(\theta)$$

Where $f(t)$ is a fictitious force-moment acting on the end-effectors of the robot arm, $M_\chi(\theta)$ represents the mass matrix of the distributed robotic joint in Cartesian space, $\chi(\theta(t))$ is an appropriate Cartesian vector representing position and orientation of the end-effectors in Cartesian space. $V_\chi(\theta, \dot{\vartheta})$ is the Coriolis term of the robotic system in Cartesian space, $G_\chi(\theta)$ is the gravity term of the robotic system in Cartesian space. θ is the angular position of motor, the vector of joint angles of the robotic arm. $\dot{\vartheta}$ is the angular velocity of the motor, $\ddot{\theta}$ is the angular acceleration of the motor.

The advanced fuzzy self-tuning PID controller provide a current/voltage control output variable v_{ctrl} to a motor driver, the different motor drivers have been designed in this control systems, for instance, a switch-mode (chopper), constant-current driver with multiple channels is designed in one control unit; its current control inputs are low current, high impedance inputs, which allows the use of un-buffered DAC or external high resistive resistor divider network. Each driver in the control system contains a clock oscillator, which is common for all the driver channels, a set of comparators and flip-flops implementing the switching control, and two output H-bridges for each motor, including recirculation diodes. Maximum output current is controlled at 750mA per channel. The DSP scales and then generates PWM using the 68376's TPU from the control output $V_{ctrl}(t)$, The velocity control is achieved through varying the voltage across the terminals of a motor by the Pulse Width Modulation that is the continuous fast switching of motor voltage. By varying the duty cycle from 0% to 100%, the effective voltage across a motor can be established from a set input of PMW duty cycle (V_{motor}). The PWM duty cycle v_{motor} is fed into the motor drive to drive the DC servo motors; the motors output the force $f(t)$, which support and control the robotic operations in robotic Cartesian based control space including robotic displacements and robotic rotations.

The robotic torqueses come from the input force $f(t)$. In Cartesian robotic task space, it can be represented as

$$\tau = J^T(\theta) \times f(t)$$

Here $J(\theta)$ are Jacobians, a time-varying linear transformations, $J^T(\theta)$ is the transpose Jacobian transformation; only once in the case of a strictly Cartesian robot arm, we can simplify the $J^T(\theta)$ to the Jacobian's inverse transformation $J^{-1}(\theta)$.

The voltage output $\upsilon(t)$ of the motor driver and motor current $i(t)$ supplied by motor driver, input to the robotic motors, the transfer function from the driver input to the robotic force is $G_i(\tau(s))$.

$$G_i(s) = \frac{f(s)}{\upsilon(s)} = f_i(\tau(t), \upsilon(t))$$

The desired control voltage V_{ctrl}, supplied by the advanced Fuzzy self-tuning PID robotic controller, consists of two components: robotic motion profile voltage control component $u_{profile}(t)$ and the robotic force/torques/current voltage Fuzzy control component $u_{force/torque}(t)$; i.e. the combined control input is: $V_{ctrl}(t) = (\alpha\ u_{profile}(t) + \beta\ u_{force/torque}(t))$

Where α and β are forward control gains coefficients. In the profile nonlinear fuzzy self-tuning PID controller, the nonlinear fuzzy control

algorithm is implemented on the control DSP, based on the Fuzzy control algorithm, the desired control gains K_p(t), K_i(t), *and* K_d(t) for the motors are sel-tuning on line in the control systems, and thus, the equivalent control components of the fuzzy PID controller are varied on line, it can be represented as follows:

$u_{profile}(t) = K_p(t)\Delta U(t) + K_i(t)\int \Delta U(t)dt + K_d(t)\frac{d(\Delta U(t))}{dt}$

Here:

Proportional gain control is $Kp(t)\Delta U(t)$

Integral gain control is $Ki(t)\int \Delta U(t)dt$

Derivative gain control is $Kd(t)\frac{d(\Delta U(t))}{dt}$

There is a RRO (relative rate observer) to estimate the error derivative $dE(t)$, and construct $\Delta dE(t)$, the derivative of $dE(t)$.

Here the motion profile control error is:

$$E(t) = E_p(t) = \chi_d(t) - \chi(t)$$

And the RRO outputs the control variable $u_{RRO}(t)$ to the PID parameter regulators; such that regulator produce the regulation control output $\gamma_{selfTuning}(t)$, which make the PID have a best-performance control gains for the robotic systems.

3 NONLINEAR FUZZY CONTROLLER

As shown in Figure 3, the error input for the motion profile is

$$E_p(t) = \chi_d(t) - \chi(t)$$

The error input for the force profile is:

$$E_F(t) = F_d(t) - F(t)$$

The desired voltage control of the nonlinear Fuzzy PID is:

$$V_{ctrl}(t) = (\alpha\ u_{profile}(t) + \beta\ u_{force/torque}(t))$$

This control output variables from the advance Fuzzy control PID control the robotic systems, it has achieved a high-performance actual profile in robotic Cartesian space. The fuzzy control is not only responsible to regulate the PID control gains, it also provide the control variables for the current/force/torque control closed-loop. The Fuzzy control principle is show in Figure 4.

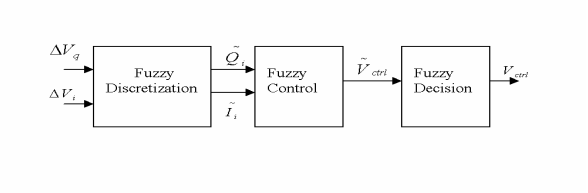


Figure 4: Nonlinear Fuzzy Controller.

Where the time continuous variables ΔV_q and ΔV_i are the whole set of the control errors in the control systems; which are the error set of $\{E_p(t), E_F(t)\}$.

Basically, the nonlinear fuzzy controller consists of three parts: the fuzzy discretization by use of the fuzzy membership functions, the fuzzy control based on the fuzzy control rules, and fuzzy decision through the weighting meaning calculation.

4 CONCLUSIONS

The real-time performance results demonstrates that the maximum Euler distance error of the Fuzzy self-tuning PID equals to 0.0005239 inches, the PID control accuracy is 0.175565 inches. The control accuracy using a Fuzzy self-tuning PID controller is improved compared a traditional PID controller in Cartesian space

$$\{Xaxis, Yaxis, Z_1axis, Z_2axis, \mu_1(\theta), \mu_2(\theta)\}.$$

REFERENCES

Shin, K. G: "Trajectory Planning for Robot Control: A Control System Perspective, Control and Dynamic System Series, Advances in Robotics System, " *Academic Press Inc.1991*, pp. 105–146.

Yasuhiko Dote, *Servo motor and motion control using signal processors,* TJ216.D68, Prentice Hall,: Englewood Cliffs, N.J. 1990.

Bill Drury, *The control techniques drives and controls handbook,* TK4058.D78, Stevenage: Institue of Electrical Engineers, 2001.

Sakae Yamamura, *AC motors for high-performance applications: analysis and control*, TK2791.Y35, New York: Dekker, 1986.

Ganwen Zeng and Ahmad Hemami, "An overview of robot force control," *Robotica*, 1997, pp. 473-482.

IDENTIFICATION OF MODELS OF EXTERNAL LOADS

Yuri Menshikov
Dnepropetrovsk University, Nauchnaja st.13, 49050 Dnepropetrovsk, Ukraine
du-mmf@ff.dsu.dp.ua

Keywords: External load, models, identification.

Abstract: In the given work the problem of construction (synthesis) of mathematical model of unknown or little-known external load (EL) on open dynamic system is considered. Such synthesis is carried out by special processing of the experimentally measured response of dynamic system (method of identification). This problem is considered in two statements: the synthesis of EL for single model and the synthesis of EL for models class for the purposes of mathematical modelling. These problems are ill-posed by their nature and so the method of Tikhonov's regularization is used for its solution. For increase of exactness of problem solution of synthesis for models class the method of choice of special mathematical models (MM) is used. The calculation of model of external load for rolling mills is executed.

1 INTRODUCTION

At mathematical modeling of real motion of open dynamic systems is important the correct choice of mathematical model of external load on system. The most accessible information about EL is contained in reactions of object on these loads which can be measured experimentally simply enough from the technical point of view. The determination problem of size and character of change of EL based on results of experimental measuring of object responses has been called the problem of identification of EL (Gelfandbein and Kolosov,1972), (Ikeda, Migamoto and Sawaragi, 1976), (Menshikov, 1983). Such approach has some advantages: the construction of model of EL is being carried out on basis of objective information (experimental measuring); the results of mathematical modeling later on are good even in case of the great inaccuracy of MM.

At construction of mathematical model of concrete dynamic system the different authors use the various simplifying assumptions. The whole set of possible equivalent mathematical models of real object (dynamic system) are being obtain in result (Menshikov, 1985). So it will be useful to build the common model of EL which is the best in some sense for class of possible mathematical models of real object (Menshikov, 1985). The statement of such problem can have application in mathematical modeling, systems control, in detection of faults and so on.

2 STATEMENT OF PROBLEM

We shall suppose for simplicity that with the aid of known internal interactions (for example, measured experimentally) some subsystem of initial dynamic system can be received at which is known one variable status and all external loads except the external load which is being investigated. If at a subsystem two variable statuses are known, then this subsystem is being replaced with more simple subsystem, at which one known variable status executes a role of known external load (Menshikov, 1983,1985,1994).

Let us suppose that the motion of the received open dynamic subsystem is being described by system of the linear ordinary differential equations with constant coefficients

$$\dot{\overline{X}} = B\,\overline{X} + C\,\overline{Z}\ ,$$

$$\overline{Y} = D\,\overline{X} + F\,\overline{Z}\ ,$$

where $\overline{X} = (x_1, x_2, ..., x_n)^*$, $\overline{Z} = (z_1, z_2, ..., z_{n_1})^*$, $\overline{Y} = (y_1, y_2, ..., y_{n_2})^*$ ((.)* is the mark of transposition); $z_1 = z$ is the researched external load; B, C, D, F are matrixes with constant coefficients appropriated dimensions, moreover D is

diagonal matrix containing only one not zero element, F is diagonal matrix containing only first zero element, $\overline{X}$ is the vector - function of status variables, $\overline{Y}$ is the vector - function of observed variables.

A problem of determination of scalar model z(t) of EL in many cases can be reduced to the solution of the linear integral equation Volltera of the first kind (Menshikov, 1983,1985,1994)

$$\int_0^t K(t-\tau)\,z(\tau)\,d\tau = u_\delta(t),$$

$$\text{or } A_p\,z = u_\delta,\ z \in Z,\ u_\delta \in U; \qquad (1)$$

where Z,U are B- functional spaces, $A_p : Z \to U$.

The function u_δ is obtained from experiment with a known error δ:

$$\left\| u_T - u_\delta \right\|_U \le \delta,$$

where u_T is an exact response of object on real external load.

We denote by $Q_{\delta,p}$ the set of functions which satisfy the equation (1) with the exactness of experimental measurements with a fixed operator A_p:

$$Q_{\delta,p} = \{z : z \in Z, \left\| A_p\,z - u_\delta \right\|_U \le \delta\}.$$

The set of $Q_{\delta,p}$ is unbounded set in norm of space U as A_p is a compact operator (Tikhonov, Arsenin , 1979). Any function from $Q_{\delta,p}$ is the good mathematical model of external load. However not all of them are convenient for further use in mathematical modeling. Let the value some continuous non-negative functional $\Omega[z]$, defined on Z_1 (Z_1 is everywhere dense set in Z) characterizes a degree of use convenience of functions from the set $Q_{\delta,p}$.

Let the function $z_p \in Q_{\delta,p}$ satisfies the condition:

$$\Omega[z_p] = \inf_{z \in Q_{\delta,p} \cap Z_1} \Omega[z]. \qquad (2)$$

Function z_p we shall name as *the solution of synthesis problem of EL model*.

Furthermore there are no reasons to believe that the function z_p will be close to real external load. It is only convenient model of external load to use for mathematical modeling later.

Let the operator A_p depends on vector-parameters of mathematical model $p = (p_1, p_2, \ldots, p_m)^*$, $p \in R^m$. It is supposed that the parameters of mathematical model are determined inexact with some error $p_i^0 \le p_i \le \hat{p}_i$, $i = 1,2,3,\ldots,m$. Therefore, the vector-parameters p can accept values in some closed domain $p \in D \subset R^m$. The operator A_p in (1) will correspond to everyone of vector-parameter $p \in D$ and they form some class of operators $K_A = \{A_p\}$.

Let's designate through h size of the maximal deviation of the operators A_p from K_A.

Let's consider the extreme problem (2) of model synthesis of EL $\tilde{z} \in Q_{\delta,h}$ for class of models K_A [2,3,4]. The set of the possible solutions for all A_p has the following form in this case:

$$Q_{\delta,h} = \{z : z \in Z, A_p \in K_A, \left\| A_p\,z - u_\delta \right\|_U \le \delta + \left\| z \right\|_Z\}. \qquad (3)$$

Any function from $Q_{\delta,h}$ brings about the response of mathematical model, which coincides with the response of real object with an error, which takes into account an error of experimental measurements and error of a possible deviation of parameters of a vector $p \in D$. A problem of finding of mathematical model $\tilde{z} \in Q_{\delta,h}$ of external load with is convenient for use later was called by analogy to the previous problem by *a problem of models synthesis for a class of models* (Menshikov, 1985).

The set of the solutions of inverse problem of synthesis with fixed operator A_p from K_A contains elements with unlimited norm (incorrect problem), therefore size $\delta + \left\| z \right\|_Z$ can be indefinitely large. Formally such situation is unacceptable, as it means, that the error of mathematical modeling is equal to infinity, if as models to use any function from $Q_{\delta,h}$. Hence not all functions from $Q_{\delta,h}$ will be "good" models of EL.

Further we shall believe, that the size $\left\| u_\delta \right\|_U$ exceeds an error of experimental measurements δ, i.e. $\delta < \left\| u_\delta \right\|_U$. Otherwise the zero element of space Z belongs to set $Q_{\delta,h}$ with any operator $A_p \in K_A$, for which $A_p 0 = 0$. This case does not represent practical interest, as the response u_δ can be received with trivial model of EL.

Let's consider the union of sets of the possible solutions $Q_{\delta,p}$:

$$\hat{Q} = \bigcup_{p \in D} Q_{\delta,p}, \ (\cup - \text{mark of union}).$$

As *the solution of a problem of synthesis for the class of models* z_{midl} we shall accept the *element* from $\hat{Q}$ (instead of set $Q_{\delta,h}$) $z_{midl} \in \hat{Q}$ which satisfies the condition:

$$\Omega[z_{midl}] = \inf_{z \in \hat{Q} \cap Z_1} \Omega[z].$$

For increase of exactness of problem solution of synthesis for class of models the method of choice of special MM is used (Menshikov, 1997). For the realization of such approach it is necessary to choose

within the vectors $p \in D$ some vector $p_0 \in D$ such that

$$\Omega[\, A_{p_0}^{-1} x] \le \Omega[\, A_p^{-1} x]$$

for all possible $x \in X$ and all $p \in D$. The operator A_{p_0} with parameter $p_0 \in D$ will be called the special minimal operator.

3 THE UNIFIED MATHEMATICAL MODEL OF EXTERNAL LOAD

Let's consider the problem of construction (synthesis) of EL model $z^{un} \in Z_1$ which provides the best results of mathematical modeling uniformly for all operators $A_p \in K_A$ (Menshikov and Nakonechny, 2005):

$$\left\| A_p z^{un} - u_\delta \right\|_U^2 \le \inf_{z_p} \sup_{A_b \in K_A} \left\| A_b z_p - u_\delta \right\|_U^2$$
$$\text{for all } A_p \in K_A . \qquad (4)$$

Let us name function z^{un} as *the unified mathematical model of external load for class K_A.*

Theorem. The function z^{un} exist and steady to small variations of initial data if $\Omega[z]$ is stabilizing functional.

4 IDENTIFICATION OF EXTERNAL RESISTANCE ON ROLLING MILLS

One of the important characteristics of rolling process is the moment of technological resistance (MTR) arising at the result of plastic deformation of metal in the center of deformation. Size and character of change of this moment define loadings on the main mechanical line of the rolling mill. However complexity of processes in the center of deformation do not allow to construct authentic mathematical model of MTR by usual methods. In most cases at research of dynamics of the main mechanical lines of rolling mills MTR is being created on basis of hypothesis and it is being imitated as piecewise smooth linear function of time or corner of turn of the working barrels (Menshikov, 1983,1985,1994). The results of mathematical modeling of dynamics of the main mechanical lines of rolling mills with such model MTR are different among themselves (Menshikov,1994).

In work the problem of construction of models of technological resistance on the rolling mill is considered on the basis of experimental measurements of the responses of the main mechanical system of the rolling mill under real EL (Menshikov,1983,1985,1994). Such approach allows to carry out in a consequence mathematical modeling of dynamics of the main mechanical lines of rolling mills with a high degree of reliability and on this basis to develop optimum technological modes. The four-mass model with weightless elastic connections is chosen as MM of dynamic system of the main mechanical line of the rolling mill (Menshikov, 1983,1985,1994):

$$\begin{aligned}
&\ddot{M}_{12} + \omega_{12}^2 M_{12} - \frac{c_{12}}{\vartheta_2} M_{23} - \frac{c_{12}}{\vartheta_2} M_{24} = \frac{c_{12}}{\vartheta_1} M_{eng}(t) \\
&\ddot{M}_{23} + \omega_{23}^2 M_{23} - \frac{c_{23}}{\vartheta_2} M_{12} + \frac{c_{23}}{\vartheta_2} M_{24} = \frac{c_{23}}{\vartheta_3} M_{rol}^U(t) \; ; \quad (5) \\
&\ddot{M}_{24} + \omega_{24}^2 M_{24} - \frac{c_{24}}{\vartheta_2} M_{12} + \frac{c_{24}}{\vartheta_4} M_{23} = \frac{c_{24}}{\vartheta_4} M_{rol}^L(t)
\end{aligned}$$

where $\omega^2_{ik} = c_{ik}(\vartheta_i + \vartheta_k)\vartheta_i^{-1}\vartheta_k^{-1}$, ϑ_k are the moments of inertia of the concentrated weights, c_{ik} are the rigidity of the appropriate elastic connection, M_{rol}^U, M_{rol}^L are the moments of technological resistance put to the upper and lower worker barrel accordingly, $M_{eng}(t)$ is the moment of the engine.

The problem of synthesis of MM of EI can be formulated so: it is necessary to define such external models of technological resistance on the part of metal which would cause in elastic connections of model of fluctuations identical experimental (in points of measurements) taking into account of an error of measurements and error of MM of the main mechanical line of rolling mill. Such type of problems and the methods of their solutions can find applications at construction of MM of EI in other similar situations.

The information on the real motion of the main mechanical line of rolling mill is received by an experimental way (Menshikov, 1983,1985,1994). Such information is being understood as presence of functions $M_{12}(t)$, $M_{23}(t)$, $M_{24}(t)$. Let's consider a problem of construction of models of EL to the upper working barrel. On the lower working barrel all calculations will be carried out similarly. From system (5) the equation concerning required model M_{rol}^U can be received.

$$\int_0^t \sin\omega(t-\tau)\, M_{rol}(\tau)\, d\tau = u_\delta(t)\ .$$

The size of the maximal deviation of the operators $A_T \in K_A$ was defined by numerical methods and it equal h = 0.12. An error initial data for a case Z = U = C [0,T] is equal $\delta = 0.066$ Мнм.

In figure 1 the diagrams of functions z_{midl}, z^{un} for a typical case of rolling on the smooth working barrel are submitted as solution of last equation (Menshikov, 1983,1985,1994).

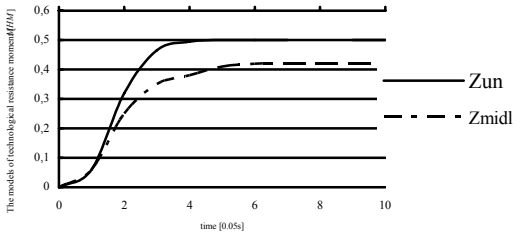


Figure 1: The diagrams of change of models of the moment of technological resistance on the rolling mill.

The results of calculations are showing that the rating from above of accuracy of mathematical modeling with model z^{un} for all $A_T \in K_A$ does not exceed 11 % in the uniform metrics with error of MM parameters of the main mechanical line of rolling mill in average 10 % and errors of experimental measurements 7 % in the uniform metrics.

The calculations of model of EL $\tilde{z}$ for a class of models K_A on set of the possible solutions $Q_{\delta,h}$ was executed for comparison. This function has the maximal deviation from zero as 0.01 Мнм.

In work [4] the comparative analysis of mathematical modeling with various known models of EL was executed. The model of load z^{un} turn out to be correspond to experimental observations in the greater degree [4].

4 CONCLUSIONS

The offered approach to synthesis of mathematical models of external loads on dynamical system can find application in cases when the information about external loads is absent or poor and also for check of hypotheses on the basis of which were constructed the known models of external loads.

REFERENCES

Gelfandbein Ju.K., Kolosov L.V., 1972. *Retrospective identification of impacts and hindrances*, Moscow, Science.

Ikeda S.I., Migamoto S., Sawaragi Y., 1976. Regularization method for identification of distributed systems. In *Proc. Identification and evaluation of parameters of systems. IY a Symposium IFAC*, Tbilisi, USSR, 1976, vol.3, Preprint. Moscow.

Menshikov Yu.L., 1983. About a problem of identification of external loads on dynamical objects as a problem of synthesis. In *J. Dynamics and strength of heavy machines*, Dnepropetrovsk University, vol.7, Dnepropetrovsk, Ukraine.

Menshikov Yu.L., 1994. The Models of External Actions for Mathematical Simulation. In: *System Analysis and Mathematical Simulation (SAMS)*, New-York, vol.14, n.2.

Menshikov Yu.L., 1985. The synthesis of external load for the class of models of mechanical objects. In *J. of Differential equations and their applications in Physics.* Dnepropetrovsk University, Dniepropetrovsk, Ukraine.

Tikhonov A.N., Arsenin V.J., 1979. *Methods of solution of ill-posed problems*, Moscow, Science.

Menshikov Yu.L., 1997. The Reduction of initial Date Inaccuracy in ill-posed problems. In *Proc. of 15th IMACS World Congress on Scientific Computation, Modelling and Applied Mathematics, Berlin, 1997, Proc. vol.6, Application in Modelling and Simulation*. Berlin.

Menshikov Yu.L., Nakonechny A.G.: Constraction of the Model of an External Action on Controlled Objects. In Journal of Automation and Information Sciences , vol.37, is. 7, (2005) 20-29.

DIGITAL PATTERN SEARCH AND ITS HYBRIDIZATION WITH GENETIC ALGORITHMS FOR GLOBAL OPTIMIZATION

Nam-Geun Kim, Youngsu Park and Sang Woo Kim
Division of Electrical and Computer Engineering, Pohang University of Science and Technology (POSTETH), Pohang, Korea
kimng@postech.ac.kr, youngsu@postech.ac.kr, swkim@postech.ac.kr

Keywords: Global optimization, Genetic algorithms, Pattern search.

Abstract: In this paper, we present a new evolutionary algorithm called genetic pattern search algorithm (GPSA). The proposed algorithm is closely related to genetic algorithms (GAs) which use binary-coded genes. The main contribution of this paper is to propose a binary-coded pattern called digital pattern which is transformed from the real-coded pattern in general pattern search methods. In addition, we offer a self-adapting genetic algorithm by adopting a digital pattern that modifies the step size and encoding resolution of previous optimization procedures, and chases the optimal pattern's direction. Finally, we compare GPSA with GA in the robustness and performance of optimization. All experiments employ the well-known benchmark functions whose functional values and coordinates of each global minimum have already been reported.

1 INTRODUCTION

Global optimization has attracted much attention recently (Horst and Pardalos, 1995; Pardalos et al., 2000; Pardalos and Romeijn, 2002), because of a wide spectrum of applications in real-world systems. Global optimization refers to finding the extreme value of a given function in a certain feasible region, and such problems are classified in two classes; unconstrained and constrained problems. This paper concerns a class of optimization algorithms that can be applied to bound constrained problems

$$\begin{aligned} \min \quad & f(x) : \mathbf{R}^n \to \mathbf{R}, \\ \text{subject to} \quad & x \in \mathbf{R}^n, \\ & l_i \le x_i \le u_i, \;\; i = 1, \cdots, n, \end{aligned} \tag{1}$$

where l_i, $u_i \in \mathbf{R}$ and $l_i < u_i$.

Although the works that deal with the global optimization are still not enough, they manage to confront the rapid growth of applications. And such work have yielded new practical solvers for global optimization, called meta-heuristics. The structures of meta-heuristics are mainly based on simulating nature and artificial intelligence tools (Osman and Kelly, 1996). Genetic algorithms (GAs) are one of the most efficient meta-heuristics (Goldberg, 1989; Michalewicz, 1996), that have been employed in a large variety of problems. However, most meta-heuristics including GAs suffer from slow convergence that brings about heavy computational costs mainly because they may fail to detect promising search directions, especially in the vicinity of local minima owing to their random constructions.

Combining meta-heuristics with local search methods is a practical solution in overcoming the drawbacks of slow convergence and random constructions of meta-heuristics. In these hybrid methods, local search strategies are included inside meta-heuristics to guide them in the vicinity of local minima, and to overcome their slow convergence especially in the final stage of the search. This paper pursues that approach and proposes a new hybrid algorithm that combines GAs with a new pattern search method. Pattern search methods are a class of direct search methods that require neither explicit nor approximate derivatives. Abstract generalizations of pattern search methods have been provided in (Torczon, 1997; Audet and Dennis, 2003). We will adopt a new idea in pattern search to form a hybrid algorithm. The new pattern search method, called digital pattern search (DPS) method, digitizes the patterns of pattern search methods into binary-codes. Thus, we

can easily combine GAs and pattern search method to construct a global search method called genetic pattern search algorithm (GPSA).

There have been some attempts to utilize the idea of hybridizing local search methods with GA. Simple hybrid methods use the GAs or local search methods to generate the points for new population and then apply other techniques to improve this new population (Günal, 2000; Zentner et al., 2001). Other hybrid methods do some modifications in the GA operations; selection, crossover and mutation using local search methods (Musil et al., 1999; Yang and Douglas, 1998; Yen et al., 1998; Hedar and Fukushima, 2004). However, the method proposed in this paper is different from these hybrid methods in many aspects. One of the main differences lies in the coding representation. We use the DPS methods in which digital patterns are binary-coded genes, and it is capable of using the evolutionary operators in GAs without modifications. Another significant difference is the self-adapting genetic algorithms that modify the step size and chase the approximate optimal direction by using local information from digital patterns. Numerical results from well-known benchmark functions indicate that GPSA exhibits a very promising performance in obtaining the global minima of multimodal functions.

In the remainder of the paper, we briefly review the basics of GAs and pattern search methods in Section 2. Section 3 proposes the DPS methods. The description of the main GPSAs is given in Section 4. In Section 5, we show experimental results. Finally, the conclusion is given in Section 6.

Notation. Let **B**, **R**, **Q** and **Z** denote the sets of binary, real, rational and integer numbers, respectively. All norms will be Euclidean vector norms or the associated operator norm.

2 BACKGROUND

In this section, we will give a brief description of GAs and pattern search methods. Both of them only use the function values rather than derivatives, and they can be used for problems with discrete design parameters. However, they are different in the coding representation. GAs use binary-coded genes, while pattern search methods use real-coded (floating-point) genes. We propose a digital pattern in order to hybridize GAs and pattern search methods.

2.1 Genetic Algorithms

GAs are algorithms that operate on a finite set of points, called a *population*. The population consists of the m bit string $s_{i,m} = [b_{i,m}, \cdots, b_{i,1}]$, where $b \in \mathbf{B}$ and $i \in \{1, \cdots, n\}$, which can be interpreted as the encoding of a vector $x \in \mathbf{R}^n$ for problem (1).

GAs are derived on the principles of natural selection and they incorporate operators for fitness assignment, selection of points for recombination, recombination of points, and mutation of a point.

The pseudo code in Figure 1 describes the steps executed in a general GA.

Randomly generate an initial population $P(0) := \{S_1(0), \cdots, S_\mu(0)\}$ where $S(t) = [s_{n,m}, \cdots, s_{1,m}]$.
Determine the fitness of each individual.
Repeat $t = 1, 2, \cdots$
  Perform recombine with probability p_r.
  Perform mutation with probability p_m.
  Determine the fitness of each individual.
  Perform replacement with an elitist replacement policy.
Until some stopping criterion is satisfied.

Figure 1: Pseudo code of general GA.

GAs start by generating an initial population $P(0)$ of μ randomly generated points $S(0)$. Then, the fitness values are evaluated for each point in $P(0)$. The fitness of a point indicates the worth of the point in relation to all other points in the population. The selected points are recombined to a new pair of points. In recombination, the crossover position is randomly selected with a probability of $p_r \in [0,1]$ and the bits after this position are exchanged between the two points. Each recombined point is mutated by a mutation, which changes the value of some bits of the binary strings with a probability of $p_m \in [0,1]$. Afterwards, replacement selects the μ_e fittest points ($0 < \mu_e < \mu$) of the generation as the elite set. These points will be put in the next generation.

2.2 Pattern Search Methods

According to (Audet and Dennis, 2003), pattern search methods have common things after a finite number of iteration. They search for a cost function value lower than that of the current iterate x_k on the trial points in the *poll set*

$$L_k = \{x_k + \Delta_k p_k,\ p_k \in P_k\}, \tag{2}$$

where $\Delta_k > 0$ is a step size, and a *pattern* p_k is the columns of the *pattern matrix* P_k defined in (Torczon, 1997). The pattern matrix is decomposed into a basis matrix $B \in \mathbf{R}^{n\times n}$ and a generating matrix $C_k \in \mathbf{Z}^{n\times p}$, $p > 2n$. Restrictions on C_k guarantee that the columns of BC_k span $\mathbf{R}^n$. Conceptually, the generating matrix

defines the search directions, while the basis matrix rotates and scales the search directions to determine the coordinate system used during the search.

In addition, each PS method has a rule called *search step* (Audet and Dennis, 2003) that selects a finite number of points on a *mesh* defined by

$$M_k = \{x_k + \Delta_k P_k z,\ z \in \mathbf{Z}^p\}. \quad (3)$$

At iteration k, the mesh is centered around the current iterate x_k, and its fineness is parameterized through the step size Δ_k. The search step strategy that gives the set of points is usually provided by the user; it must be finite and the set can be empty.

The pseudo code in Figure 2 describes the main elements of a pattern search method. It is based on the method presented in (Audet and Dennis, 2003).

Let the initial solution $x_0 \in \mathbf{R}^n$ and step length Δ_0 be given.
Repeat $k = 1, 2, \cdots$
  Perform the **Search Step**:
    Evaluate f on a finite subset of trial points on the mesh M_k defined by (3).
  Perform the **Poll Step**:
    Evaluate f on the poll set defined by (2).
  Update the pattern matrix and Δ_k.
Until some stopping criterion is satisfied.

Figure 2: Pseudo code of pattern search method.

The scenario of a pattern search method starts with fitting the initial solution, and then two search stages are invoked. The first staged is a *search step* in which any search procedure can be defined by the user to generate trial points from M_k. The main role of the *search step* is to achieve faster convergence of pattern search method. The other stage called *poll step* is performed as a systematic search in order to exploit a region around the current solution. If the *search step* and *poll step* fail to produce a trial step that gives a simple decrease, then the step size is reduced to refine the mesh. Otherwise, the step size is increased or preserved. The pattern search method may be terminated when the step size becomes small enough.

3 DIGITAL PATTERN SEARCH METHOD

In this section, we propose the digital pattern search (DPS) method before introducing genetic pattern search algorithm (GPSA). This section formulates the abstraction of DPS methods. The definitions and the algorithm for generalized DPS methods follow descriptions of pattern search methods provided in (Torczon, 1997) and (Audet and Dennis, 2003).

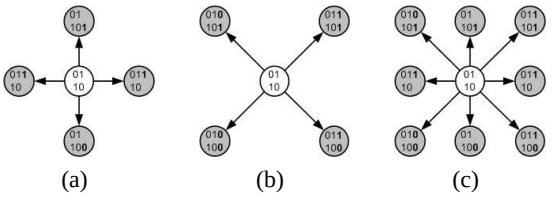


Figure 3: Trial points generated by some kinds of digital patterns. Ash-colored bulbs are trial points. (a), (b), and (c) correspond to digital patterns of compass search, evolutionary operation using factorial designs and coordinate search, respectively.

3.1 Digital Pattern

Adding 0 or 1 to an existing binary string as a least significant bit (LSB) can be interpreted as generating trial points of pattern search methods. The decoded real number of a new binary string that is appended LSB "0" is decreased and that is appended LSB "1" is increased than that of the original binary string. This property is adopted as pattern to generate trial points which are solution candidates. Figure 3 depicts the 2 bit string's trial points that are presented by binary strings or binary matrices in which each row represents an encoded string for the corresponding parameter.

Pattern search methods can be divided by pattern matrix P_k into a compass search, evolutionary operation, coordinate search, and so on (Torczon, 1997). Digital pattern can describe some kinds of patterns according to whether or not an LSB is attached to each dimension of bit strings. Figure 3 depicts some kinds of digital patterns that mimic pattern of compass search, pattern of evolutionary operation using factorial designs, and pattern of coordinate search, respectively.

The DPS method requires a mechanism for decoding from each m-bit string $s_{i,m} = [b_{i,m}, \cdots, b_{i,1}]$ to the corresponding object variable $x_{i,m}$. According to the standard binary decoding function $f_d : \{0,1\}^m \rightarrow [u_i, v_i]$, where (Michalewicz, 1996), the real value is

$$x_{i,m} = f_d(s_{i,m}) = u_i + \frac{v_i - u_i}{2^m - 1} \sum_{j=0}^{m-1} b_{i,(j+1)} 2^j. \quad (4)$$

If 0 is added to the LSB of $s_{i,m}$, let the new string be termed a 0-*bit child string*, $s^0_{i,m+1}$. On the other hand, if 1 is added, it is termed a 1-*bit child string*,

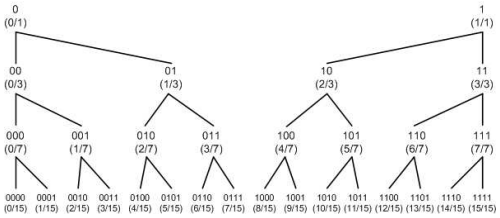


Figure 4: Biased binary tree structure together with corresponding normalized real values in parenthesis.

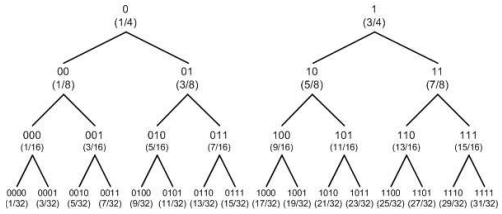


Figure 5: Unbiased binary tree structure together with corresponding normalized real values in parenthesis.

$s^1_{i,m+1}$. And the real values of the child strings are

$$x^0_{i,m+1} = \frac{u_i}{2^{m+1}-1} + \frac{2^{m+1}-2}{2^{m+1}-1}\, x_{i,m},$$
$$x^1_{i,m+1} = \frac{v_i}{2^{m+1}-1} + \frac{2^{m+1}-2}{2^{m+1}-1}\, x_{i,m}.$$

The Difference between a parent string and each child string implies the step size Δ_k in pattern search methods. For a given parent string $x_{i,m}$, the comparison between the distances of $|x^0_{i,m+1} - x_{i,m}|$ and $|x^1_{i,m+1} - x_{i,m}|$ is given by

$$| x^1_{i,m+1} - x_{i,m} | - | x^0_{i,m+1} - x_{i,m} | = \frac{(v_i - u_i) - 2x_{i,m}}{2^{m+1}-1}. \quad (5)$$

From (5), the differences between two step sizes vary according to the position of the parent string in the finite intervals $[u_i, v_i]$.

In Figure 4, the property in (5) is clearly visualized in the form of binary trees whose nodes are represented by binary strings and their corresponding real numbers. Owing to the specific property of digital pattern that increases the bit length of binary strings, the standard binary decoding function (4) has a biased search tendency to incline its steps toward the middle point of the finite intervals $[u_i, v_i]$ depending on the location of a parent string. Therefore, we need a more suitable binary decoding function for digital pattern and the *unbiased binary decoding function* is designed as:

$$x_{i,m} = u_i + \frac{v_i - u_i}{2^{m+1}} \left(\sum_{j=0}^{m-1} b_j 2^{j+1} + 1 \right),$$

and the real values of the child strings are given as:

$$x^0_{i,m+1} = x_{i,m} - \frac{v_i - u_i}{2^{m+2}}, \;\; x^1_{i,m+1} = x_{i,m} + \frac{v_i - u_i}{2^{m+2}}. \quad (6)$$

Both step sizes are the same, $(v_i - u_i)/\, 2^{m+2}$. Figure 5 shows that the unbiased binary decoding function guarantees the symmetric search property.

To define digital pattern, we treat bit strings with the decoded real value defined in the iterative form (6). According to (6), the real values of trial points are analogous to the description of generalized pattern search methods in (Torczon, 1997). A basis matrix can be defined a nonsingular matrix $B \in \mathbf{R}^{n \times n}$

$$B = \mathrm{diag}(l_1, \ldots, l_n) = \mathrm{diag}(u_1 - v_1, \ldots, u_n - v_n),$$

where $\mathrm{diag}(\cdot)$ is a diagonal matrix. B represents the intervals of each dimension of x. A generating matrix $C \in \mathbf{Z}^{n \times p}$, where $p > 2n$, contains in its columns combinations of $\{1, 0, -1\}$, except for the column of zeros. For example, when digital pattern for coordinate search is executed for $n = 2$, we have a generating matrix such as

$$C = \begin{bmatrix} 1 & 0 & -1 & 0 & 1 & 1 & -1 & -1 \\ 0 & 1 & 0 & -1 & 1 & -1 & -1 & 1 \end{bmatrix}.$$

It can be seen in Figure 3 (c).

Digital pattern p is then defined by the columns of the digital pattern matrix $P = BC$. Because both B and C have the rank n, the columns of P span $\mathbf{R}^n$. The step size Δ_m is defined as $\Delta_m = \frac{1}{2^m}$ under the given bit string length m. Thus the poll set composed of points neighboring the current x_m in the directions of the columns of C is expressed as

$$L_m = \{ x_m + \frac{1}{4}\Delta_m p, \; p = Bc \text{ and } c \in C \}.$$

Among them, the best one is chosen by evaluation as an optimal solution of L_m, $x^*_{m+1} = x_m + \frac{1}{4}\Delta_m B c^*_{m+1}$, where c^*_{m+1} is the column of C as a *direction vector* pointing the search direction toward the optimal solution.

3.2 Digital Step

Using the standard binary representation, the DPS method can get caught on a "Hamming cliff", being confined to the barrier of binary branches. For example, if the DPS method is started at the highest node of "0" in the tree, it can never escape from the left half plane of parameters. *Digital step* in the DPS method is employed to avoid such problems. If a binary string undergoes *increment addition* (INC)

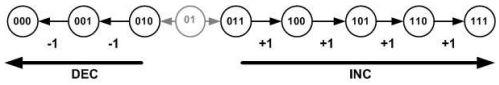


Figure 6: Process of *digital step.* in one dimensional diagram.

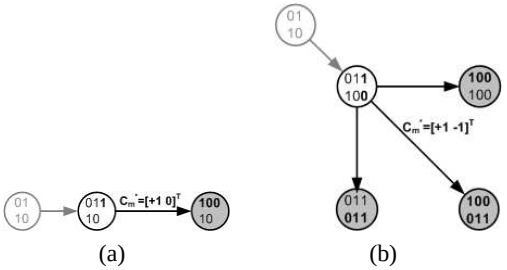


Figure 7: Process of *digital step.* in two dimensional diagram. (a) and (b) correspond to the results depended on c_m^*, $[+1\ 0]^T$ and $[+1\ -1]^T$, respectively.

or *decrement subtraction* (DEC), the real number of each processed string increases or decreases. Through the simple operations of INC and DEC for binary strings, *digital step* can readily remove the barrier between any binary trees and broadly search the effective area that has a high possibility of finding the optimal solution. Figure 6 illustrates the process of *digital step* (if direction vector is 1, then INC is executed, otherwise DEC is achieved).

In *digital step*, the objective function f is evaluated at a finite number of points on a mesh to try to find a point that yields a lower objective function value than the current point. The basic component in the definition of *digital step* is the mesh. The mesh is a discrete subset of $\mathbf{R}^n$ whose fineness is parameterized by the step size Δ_m as follows:

$$M_m = \{x_m^* + \Delta_m \mathrm{diag}(z) B c_m^* : z \in \mathbf{Z}^n\},$$

where $\mathrm{diag}(\cdot)$ is a diagonal matrix, z is the vector of nonnegative integers and c_m^* is a direction vector chosen by evaluation of poll set. This way of describing the mesh is different from the form in (Audet and Dennis, 2003). This specific sub-technique of the DPS method attempts to accelerate the progress of the algorithm by exploiting the information gained from the search.

The trial points generated by *digital step* are decided by a sort of digital patterns. Figure 7 shows the examples of *digital step* in a two dimensional search space. INC or DEC is selected by c_m^* from the previous digital pattern. Then *digital step* continues to search in the trial points which are generated by performing INC or DEC at each dimension. Generation of the trial points by the digital pattern is executed one time, while generation of the trial points by *digital step* continues until a local minimum is attained on the same bit mesh.

3.3 Digital Pattern Search Method

The pseudo code in Figure 8 describes the proposed digital pattern search method.

Set an initial row length α and a final row length β.
Randomly generate an initial binary matrix $\Gamma_{n\times\alpha}$.
for each low length $m = \alpha : \beta$ **do**
1. Perform the **Digital Pattern**.
2. Evaluate trial points and determine a direction vector.
3. **while** (a better solution is attained) **do**
   (a) Perform the **Digital Step**.
   (b) Evaluate trial points.
   end while

end for

Figure 8: Pseudo code of digital pattern search method.

The basic structure of the DPS method consists of two asynchronous loops. The outer loop (steps 1-3) selects the best trial point generated by digital pattern and hands over a direction vector to *digital step*. The inner loop (step 3) conducts finite searches in the guided direction vector until the consecutive *digital step* fails to make progress. When this occurs a local minimum of a session is found; the inner loop then terminates and the outer loop starts the next session. At steps 1 and 3(a), digital pattern and *digital step* generate $n \times m$ binary matrices as trial points, i.e., one of the matrices $\Gamma_{n\times m}$ implies that each row n represents an encoded string for the corresponding parameter, and that the row length m is exponentially proportional to the resolution of parameters. To evaluate trial points in steps 2 and 3(b), each row of Γ needs to be converted to the real number using a binary decoding function. After evaluation in step 2, the least significant column of the best trial point is appointed the direction vector which is handed over to *digital step* for further exploration.

4 GENETIC PATTERN SEARCH ALGORITHM

GPSA uses the main operations of GA; recombination, mutation, and replacement, on a population to encourage the exploration process. Moreover, the GPSA tries to improve the new children by applying

DPS method. Figure 9 shows the pseudo code describing GPSA.

```
Set an initial row length α and a final row length β.
Set an GA's generation number N.
Randomly generate an initial population P_α(0) :=
{Γ^1_{n×α}(0), ··· , Γ^μ_{n×α}(0)}
Determine the fitness of each individual.
for each low length m = α : β do
  for GA's generation k = 0 : N do
    Perform recombine with probability p_r.
    Perform mutation with probability p_m.
    Compute the fitness of each individual.
    Perform replacement with an elitist replacement policy.
  end for
  Perform the Digital Pattern.
  Compute the fitness of trial points of each individual and determine a direction vector.
  while a better solution is attained do
    Perform the Digital Step.
    Compute the fitness of trial points of each individual.
  end while
  Perform replacement with an elitist replacement policy.
end for
```

Figure 9: Pseudo code of GPSA.

The GPSA starts by generating an initial population $P(0)$ of μ randomly generated points Γ which is composed by α bit-length strings. The inner loop in GPSA incorporates GA operators: recombination, mutation, and replacement. In recombination, a crossover position is randomly selected with a probability of $p_r \in [0,1]$. Each recombined point is mutated with a probability of $p_m \in [0,1]$. Replacement selects the μ_e fittest points ($0 < \mu_e < \mu$) of the generation as the elite set. After N times iterations of the inner loop, the DPS method is applied to the points generated by the evolutionary operators and constructs each sequence of iterates that converge to a stationary point on the mesh parameterized by the step size Δ_{m+1}. The restriction on the replacement strategy ensures that the elite set is kept for further processing.

5 EXPERIMENTAL RESULTS

This section presents a performance comparison of the GPSA and the conventional GA on the well-known 8 benchmark functions whose functional values and coordinates of each global minimum are already reported in (Yao et al., 1999; Schwefel, 1995). Several kinds of benchmark functions are selected to make a generalized conclusion: functions with no local minima (f_1-f_2) and local minima (f_3-f_8). A more detailed description of each function is given in the Appendix.

5.1 Experimental Setup

For both GA and GPSA tests, we used a population size μ of 10 with the elite set $\mu_e = 1$, and for each problem the number of trials was 100. The recombination operator was uniform crossover with a probability $p_c = 0.5$, and the mutation probability was $p_m = 0.001$. The length per object variable in GA was 10, and in GPSA, the initial and final low length per object variable were $\alpha = 3$ and $\beta = 10$, respectively. The number of generations was $4,000$ in GA and $N = 50$ in GA loop of GPSA.

To implement GPSA, it is necessary to determine a proper kind of the generating matrix in digital pattern. We used the standard $2n$ directions, $C = \{e_1, \cdots, e_n, -e_1, \cdots, -e_n\}$, where $e_i \in \mathbf{R}^n$ is the $i-th$ unit vector, because it gives a linear increase of function evaluation with problem dimension.

GA was terminated after $40,000$ function evaluations, and the performance comparisons between GPSA and GA were made based upon the termination point of GPSA. Our experimental analysis considers three performance measures: the number of trials which succeed in attaining to the global optimum for each benchmark function, the number of cost function evaluations during simulations, and the value of the best solution found.

5.2 Numerical Results

Figure 10 shows the performances of GPSA and GA on the benchmark functions. The results of GPSA were selected through 100 independent trials as the best case and the worst case, and the result of GA was averaged over 100 independent trials. The GA converged faster than the GPSA for most functions initially, around 4,000 to 6,000 of function evaluations. However, GPSA overperformed GA obviously while fewer function evaluations. Although GA quickly approaches the neighborhood of the global minimum, GA has a difficulty in obtaining some required accuracy. DPS method's ability to accelerate the search and to refine the solution more evenly helps GA in achieving good performance.

To judge the success of a trial, we used the condition

$$|f^* - \hat{f}| < \varepsilon_1 |f^*| + \varepsilon_2,$$

Table 1: Results of GPSA. The results were averaged over 100 independent trials where "SUCC %" indicates the ratio of trials which succeed in attaining the global optimum , "EVAL #" means the average number of function evaluations, and "VAR" means the variance of trials which succeed. Functions (Yao et al., 1999; Schwefel, 1995): SP (spare function), SC1 (schwefel's problem 2.22), SC2 (schwefel's problem 1.2), SC3 (schwefel's problem 2.26), GR (griewank function), AC (ackley function), RA (rastrign function), and SH (shubert function). And "n" is the number of variables.

Function	SUCC %	EVAL #	VAR
$SP\ (n=10)$	100	5504.35	0.0
$SC1\ (n=10)$	100	4773.82	0.0
$SC2\ (n=10)$	44	5635.07	0.0
$SC3\ (n=2)$	92	2116.56	0.0
$GR\ (n=10)$	100	5321.53	0.0
$AC\ (n=10)$	100	5483.51	0.0
$RA\ (n=10)$	92	7590.75	0.0
$SH\ (n=2)$	38	4240.23	0.0013

where $\hat{f}$ refers to the best function value obtained by GPSA, f^* refers to the known exact global minimum, and ε_1 and ε_2 are small positive numbers. We set ε_1 and ε_2 equal to 10^{-3} and 10^{-6}, respectively. The results are shown in Table 1, where the average number of function evaluations and the variance are related only to successful trials. Table 1 shows that GPSA reached the global minima in a very good success rate for the majority of the tested functions. Moreover, the numbers of function evaluations and the average errors show the efficiency of the method.

On the other hands, GA had few successful trials on any test functions at the termination point of GPSA.

6 CONCLUSIONS

This paper first developed a new class of pattern search method that digitizes the patterns, called the digital pattern search (DPS) method. Then, we presented a new hybrid global search algorithm, the genetic pattern search algorithm (GPSA), which has a self-adapting technique to modify the step size and chase the approximate optimal direction. Applying the DPS method in addition to the ordinary GA operators such as recombination and mutation enhances the exploration process and accelerates the convergence of the proposed algorithm. The experimental results also showed that the GPSA works successfully on some well known test functions.

REFERENCES

Audet, C. and Dennis, Jr., J. E. (2003). Analysis of generalized pattern searches. *SIAM J. on Optim.*, 13(3):889–903.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Boston, MA.

Günal, T. (2000). A hybrid approach to the synthesis of nonuniform lossy transmission-line impedance-matching sections. *Microwave and Optical Technology Letters*, 24:121–125.

Hedar, A. and Fukushima, M. (2004). Heuristic pattern search and its hybridization with simulated annealing for nonlinear global optimization. *Optim. Methods and Software*, 19:291–308.

Horst, R. and Pardalos, P. M. (1995). *Handbook of Global Optimization*. Kluwer Academic Publishers, Boston, MA.

Michalewicz, Z. (1996). *Genetic algorithms + data structures = evolution programs*. Springer-Verlag, London, UK.

Musil, M., Wilmut, M. J., and Chapman, N. R. (1999). A hybrid simplex genetic algorithm for estimating geoacoustic parameters using matched-field inversion. *IEEE J. Oceanic Eng.*, 24(3):358–369.

Osman, I. H. and Kelly, J. P. (1996). *Meta-Heuristics: Theory and Applications*. Kluwer Academic Publishers, Boston, MA.

Pardalos, P. M. and Romeijn, H. E. (2002). *Handbook of Global Optimization*. Kluwer Academic Publishers, Boston, MA.

Pardalos, P. M., Romeijn, H. E., and Tuy, H. (2000). Recent developments and trends in global optimization. *J. Comput. Appl. Math.*, 124(1-2):209–228.

Schwefel, H.-P. (1995). *Evolution and Optimum Seeking: The Sixth Generation*. Addison-Wesley, New York, NY.

Torczon, V. (1997). On the convergence of pattern search algorithms. *SIAM J. on Optim.*, 7(1):1–25.

Yang, R. and Douglas, I. (1998). Simple genetic algorithm with local tuning: efficient global optimizing technique. *J. Optim. Theory Appl.*, 98(2):449–465.

Yao, X., Liu, Y., and Lin, G. (1999). Evolutionary programming made faster. *IEEE Trans. on Evol. Comput.*, 3(2):82–102.

Yen, J., Liao, J., Randolph, D., and Lee, B. (1998). A hybrid approach to modeling metabolic systems using a genetic algorithm and simplex method. *IEEE Trans. on Syst., Man, and Cybern. B*, 28(2):173–191.

Zentner, R., Sipus, Z., and Bartolic, J. (2001). Optimization synthesis of broadband circularly polarized microstrip antennas by hybrid genetic algorithm. *Microwave and Optical Technology Letters*, 31:197–201.

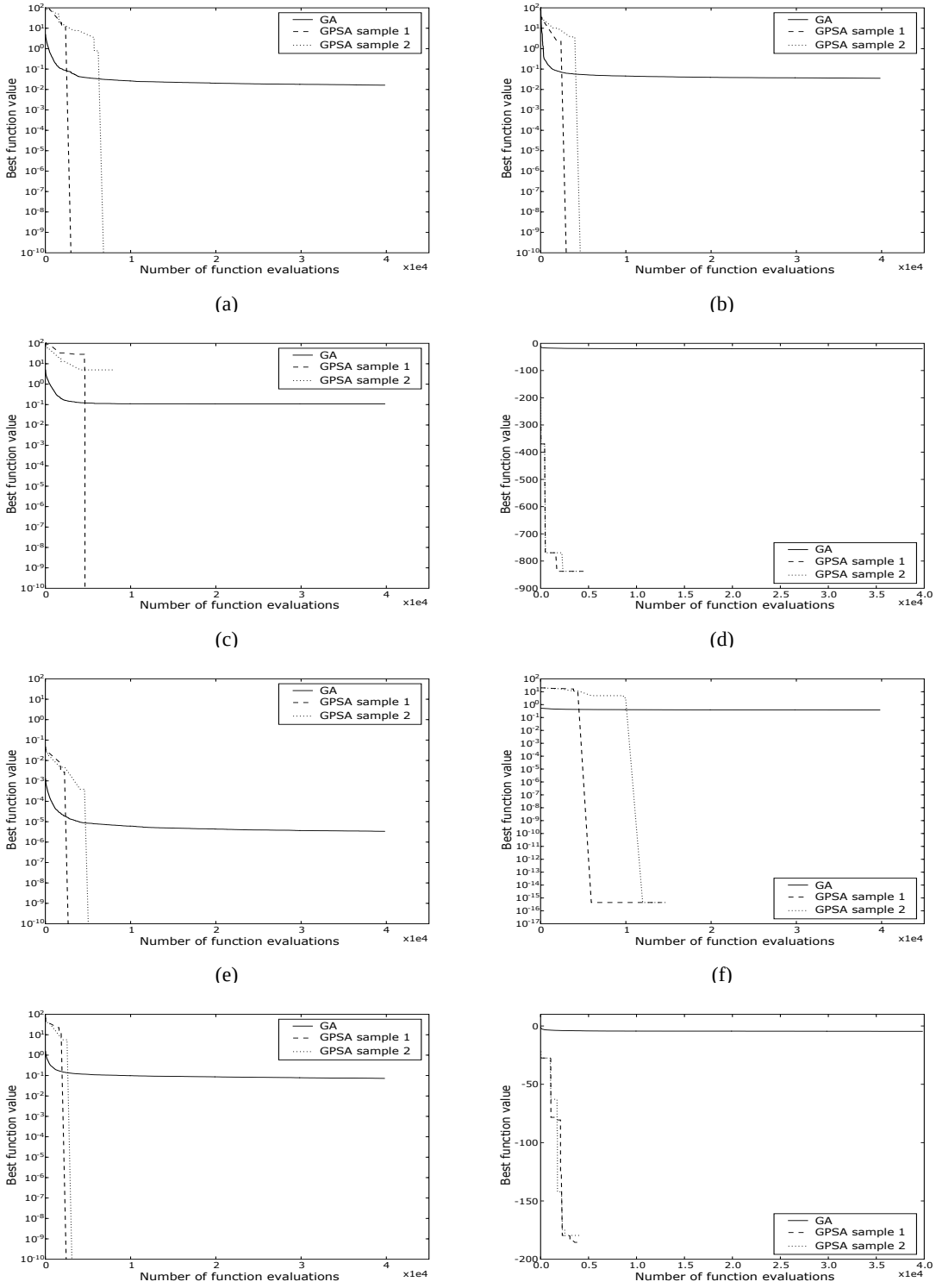


Figure 10: The comparisons of the performance between GPSA and GA. The results of GPSA were selected through 100 independent trials as the best case and the worst case, and the result of GA was averaged over 100 independent trials. (a)-(h) correspond to the results of test functions f_1-f_8, respectively.

MODELING WITH CURRENT DYNAMICS AND VIBRATION CONTROL OF TWO PHASE HYBRID STEPPING MOTOR IN INTERMITTENT DRIVE

Ryota Mori, Yoshiyuki Noda, Takanori Miyoshi, Kazuhiko Terashima
Department of Production Systems Engineering, Toyohashi University of Technology
Hibarigaoka 1-1, Tempaku, Toyohashi, 441-8580, Japan
{*mori,noda, miyoshi, terasima*}*@syscon.pse.tut.ac.jp*

Masayuki Nishida, Naohiko Suganuma
Tokyo Weld Co., Ltd.
Ashitaka 292-50, Numazu, Shizuoka, 410-0001, Japan
{*m-nishida,n-suganuma*}*@tokyo-weld.co.jp*

Keywords: Stepping motor, phase plane analysis, vibration control.

Abstract: This paper presents modeling of stepping motor and control design of input pulse timing for the suppression control of vibration. The stepping motor has the transient response of electric current for the pulse input. Therefore, the motor model considering the transient response of the current is built. The validity of the proposed model is verified by comparing the model considering the transient response of the current with the one without its consideration. Design of the pulse input timing in the method of the four pulse drive is realized to achieve the desired angle without vibration and overshoot using an optimization method. Finally, the effectiveness of the proposed method is demonstrated by comparing simulation results with experiments.

1 INTRODUCTION

The stepping motor has been widely used for factory automation (FA) and office automation (OA) equipment, because it is able to realize high-accuracy positioning by an open loop control. It is used also for the production process of electronic parts, and then the settling time of the stepping motor is directly linked to the productivity. Therefore, the high speed and the low vibration are strongly desired in the production line. However, the stepping motor vibrates around the neighborhood of the equilibrium points owing to the step-wise drive from the viewpoints of the motor characteristics. To dampen the vibration of motor, (i) micro step drive method that makes changes the exciting current change in details, and (ii) a inverse phase excitation dumping, and (iii) a delay damping method, are proposed(D.Ebihara and T.Iwasa, 1984). Especially, the micro step drive is possible to drive with the low vibration. Because it is made to drive by changing at smaller angle than the basic step angle by making the exciting current change in details (D.Ebihara and T.Iwasa, 1984)D It is necessary to give the excitation instruction considering the dynamic characteristic of the system in the transient state of the start and the stop times. As an adjustment method of the excitation sequence, the method of applying a lowpass filter as a pre-compensator (T.Miura et al., 2000), and the method that uses the genetic algorithm (T.Miura and T.Taniguchi, 1999) are proposed. The vibration control considering robustness for the vibration of the inertia load is studied by these methods. Moreover, the technique for decreasing the resonance using the position and the speed feedback estimated by the observer is given (S.M.Yang and E.L.Kuo, 2003).

The stepping motor has a strong nonlinearity. Therefore, when the linear control theory is applied, it is often linearized around the equilibrium point. However, the operation area of the stepping motor is wide, so variable control gain is necessary to keep an excellent control performance. Whereas, the method using an exact linearization by means of nonlinear feedback and coordinate transformation of state space is proposed (M.Bodson et al., 1993). Moreover, the application of the control algorithm of the artificial intelligence system such as fuzzy theory (F.Betin and D.Pinchon, 1998) and neural network (K.Laid et al., 2001) are provided.

Those control methods are mostly discussed concerning with a step drive and a continuous drive. However, the high speed and the high accurate positioning system by the fine drive might be requested in the FA equipment. In this case, the positioning by a few number of pulse order is performed.

Therefore, in this paper, the vibration suppression is studied when the stepping motor is intermittently driven by a predetermined few number of step, assuming FA application. The response of the stepping motor is varied by the influence of transient response of the current, when the command pulse interval is rather short for the excitation phase. Therefore, a mathematical model comprised of the torque equation with the dynamics of the current in the generated torque is presented. By means of the phase plane analysis and the optimization for the obtained mathematical model, the pulse control timing with a low vibration is obtained. The validity of the proposed model is confirmed through the experiments and the simulations.

2 EXPERIMENTAL SETUP

The experimental setup consists of motor with the inertia load connected with a solid shaft as shown in Figure 1. The shaft has been extended from both ends of the motor, and the inertia load is installed in a one end. The encoder has been installed to the shaft on the other side.

The block diagram of the driving system of a stepping motor is shown in Figure 2. The motor driver uses the commercial item, and the motor is driven up to the fixed current drive in a full step. The pulse input is carried out by the designed timing in advance by numeric calculation. The pulse is outputted according to the timing specified from the Pulse Oscillator. The excitation current is passed by excitation phases by the motor driver, and the motor is driven. The excitation phase of the motor can be switched by the pulse being input from Pulse Oscillator to Motor Driver. In this study, the motor is used with the excitation phase comprised of Phase A, Phase B, Phase $\bar{A}$, and Phase $\bar{B}$. The excitation phase switches in order of $AB \rightarrow B\bar{A} \rightarrow \bar{A}\bar{B} \rightarrow \bar{B}A \rightarrow AB$ by two phase excitation. The rotor angle is detected by the encoder of accuracy of 36000[pulse/rev]. The encoder signal is doubled by up-down counter of four.

3 STEPPING MOTOR MODEL

3.1 Modeling of Stepping Motor

The stepping motor rotates stepwise by switching the excitation phase by input pulse. Therefore, Eq.(1) can be derived from the equation of motion of rotation system (D.Ebihara and T.Iwasa, 1984).

$$J\ddot{\theta} + D\dot{\theta} + T_L = T_M \tag{1}$$

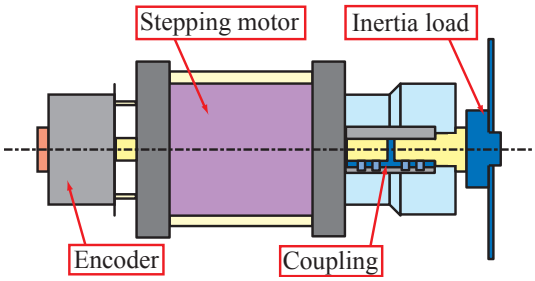


Figure 1: Construction of stepping motor.

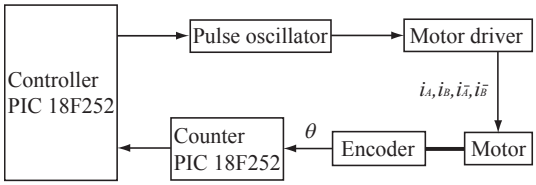


Figure 2: Block diagram of driving system for a stepping motor.

,where J is an inertia moment, D is a damping coefficient, T_L is a load torque, T_M is the generated torque of motor and θ shows a rotor angle. The generated torque can be described as the sum of the generated torque of each phase. The generated torque of each phase of two phase hybrid stepping motor is represented by the following equations.

$$T_A = -i_A K \sin(N_R\theta) \tag{2}$$

$$T_B = -i_B K \sin(N_R\theta - \frac{\pi}{2}) \tag{3}$$

$$T_{\bar{A}} = -i_{\bar{A}} K \sin(N_R\theta - \pi) \tag{4}$$

$$T_{\bar{B}} = -i_{\bar{B}} K \sin(N_R\theta - \frac{3\pi}{2}) \tag{5}$$

,where K is a torque constant, N_R is the number of a rotor teeth and, i_A, i_B, $i_{\bar{A}}$ and $i_{\bar{B}}$ are excitation current of each phase. The suffix of torque T and current i shows phase A, B, $\bar{A}$ and $\bar{B}$ respectively. Whenever the pulse is input in the two phase stepping motor, it is excited in order of $AB \rightarrow B\bar{A} \rightarrow \bar{A}\bar{B} \rightarrow \bar{B}A \rightarrow AB$. Moreover, because $i_A = -i_{\bar{A}}$ and $i_B = -i_{\bar{B}}$, then it follows that $T_A = T_{\bar{A}}$ and $T_B = T_{\bar{B}}$. Therefore the generated torque T_M becomes the sum of T_A and T_B. If a magnetic axis is defined as $\theta = 0$ when phase A and phase B are the excited states, the generated torque is shown as follows.

$$T_M = -i_A K \sin(N_R\theta + \frac{\pi}{4}) - i_B K \sin(N_R\theta - \frac{\pi}{4}) \tag{6}$$

Generally, it becomes $i_A = i_B = i$ at the fixed current drive. And Eq.(7) is derived by the trigonometric function formula.

$$T_M = -\sqrt{2}iK\sin(N_R\theta) \quad (7)$$

This formulation is explained by the reference of D.Ebihara and T.Iwasa, 1984. However, there is a transient state between the time of the pulse input and the switching time of the current of the excitation phase, as shown in Figure 3. Here, Figure 3 shows the excitation current transition of each phase after pulse input when the stepping motor is driven, and Exp I_A is a current of phase A and $\bar{A}$, Exp I_B is a excitation current of phase B and $\bar{B}$, Sim I_A and Sim I_B is the assumed current in simulation. From Figure 3, when the first pulse is input, the Exp I_A is switched from the plus into the minus. It is meant to change the phase $\bar{A}$ from the phase A, because Exp I_A is the excitation current of phase A. Here, the change of the current in Exp I_A is almost linearly changed from the value of $+1.2$[A] to the value of -1.2[A], when the excitation phase is switched. Excitation current Exp I_B of the phase B is also similar. Therefore, the excitation current is thought that the current is changed almost linearly from the input of the pulse up to the switching the excitation phase as shown in Sim I_A and Sim I_B. The current response of interval T_d[s] is shown in Eq.8, when the elapsed time after inputting the pulse is defined as T_i[s], and the time from the input of the pulse up to switching the excitation current is defined as T_d[s].

$$i_{A,B} = -(1-2\frac{T_i}{T_d})I, \qquad (0 \le T_i \le T_d) \quad (8)$$

,where I is the rated current of the motor. From Eq.(1), Eq.(6) and Eq.(8), the equation of motion is shown in Eq.(9).

$$J\ddot{\theta}+D\dot{\theta}+T_L = -i_AK\sin(N_R\theta+\frac{\pi}{4})-i_BK\sin(N_R\theta-\frac{\pi}{4}) \quad (9)$$

,where i_A is shown by the following equation. As for i_B, it is omitted in this paper, due to the limitation of paper, because it is similar equation.

$$i_A = \begin{cases} I, & (Phase\ A) \\ -(1-2\frac{T_i}{T_d})I, & (Phase\ A \rightarrow Phase\ \bar{A}) \\ -I, & (Phase\ \bar{A}) \\ (1-2\frac{T_i}{T_d})I, & (Phase\ \bar{A} \rightarrow Phase\ A) \end{cases} \quad (10)$$

3.2 Parameter Identification

The parameter of Eq.(9) is identified from the step response of stepping motor. The parameter is decided to be minimized the error sum of square by a Simplex Method(M.Hamaguchi et al., 1994). Here, J, D and T_L are the parameters including the encoder. The step response result is shown in Figure 4. The response corresponds with the experiments and the simulations, and then the identification is appropriate, because simulation agrees well with experimental results. The model parameter is shown in Table 1, where α is a step angle.

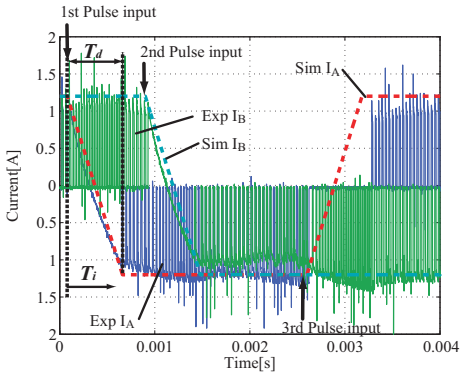


Figure 3: Current response of stepping motor.

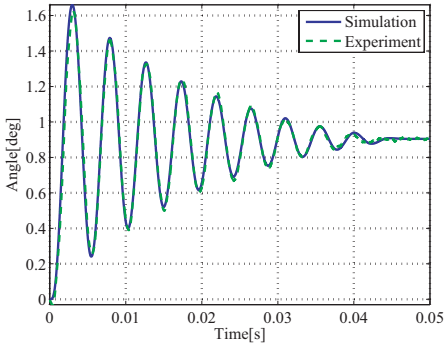


Figure 4: Step response of proposed model.

Table 1: Motor parameters.

Parameter	Value
α	0.9 [deg]
I	1.2 [A/phase]
J	164.94×10^{-7} [kg·m^2]
D	0.001442 [kg·m·s]
T_L	0.00357 [N·m]
K	0.2662 [N·m/A]
T_d	0.0007 [s]

3.3 Comparison Between Proposed and Conventional Model

The conventional model is written by the following equation using Eq.(1) and Eq.(7).

$$J\ddot{\theta} + D\dot{\theta} + T_L = -\sqrt{2}IK\sin(N_R\theta) \qquad (11)$$

The model parameters which did not consider the current response can be also identified from the step response in the same way with subsection 3.2. The parameters obtained from the optimization are shown in Table 2. The result of the step response is shown in Figure 5. Experiments and simulations are almost well in agreement as shown in Figure 5

The model which did not consider the current response is shown in Eq.(11), and the proposed model are compared. In comparison, the response of motor in experiment and simulation are checked, when four times pulses are inputted to motor. T_1[s] is the interval of input pulse from the first pulse to the second pulse. And T_2[s] is the interval of input pulse from the second pulse to the third pulse, T_3[s] is the interval of input pulse from the third pulse to the fourth pulse, and then T_1=400[μs]CT_2=2331[μs]CT_3=1428[μs]. The pulse timing of T_1, T_2, and T_3 is respectively obtained by using Simplex Method to minimize the evaluation function as shown in Eq.(12) at Section 4. Experiment and simulation results are shown in Figure 6.

The rising time of the conventional model is shorter than proposed model, and the residual vibration remains close to the desired value. However, simulation result of the proposed model agrees well with the experimental result. As a result, we confirmed that the more highly accurate model can be obtained by the proposed model considering the current response.

Table 2: Parameters of conventional model.

Parameter	Value
J	168.56×10^{-7} [kg·m^2]
D	0.001505 [kg·m·s]
T_L	0.001902 [N·m]
K	0.2582 [N·m/A]

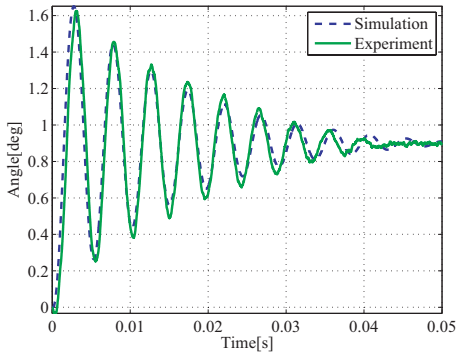


Figure 5: Step response of conventional model.

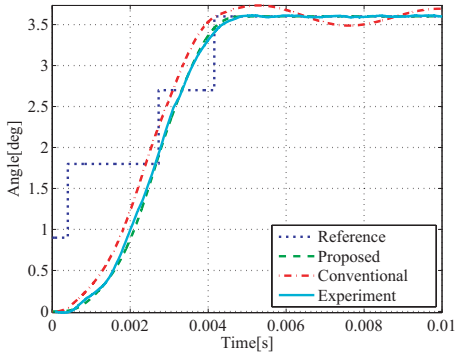


Figure 6: Comparison result of the conventional and proposed model.

4 DESIGN OF PULSE INPUT TIMING

In this paper, the time that the rotor reaches the desired angle without vibration, when input using four pulses are given to the stepping motor, is determined based on the proposed model. The time of the first input pulse is 0[s]. The one step angle is $\alpha = 0.9$[deg], so the desired angle by four input pulse is 3.6[deg]. The pulse interval T_1, T_2, and T_3 denotes the same meaning with the preceding section. When T_1 are changed from 300[μs] up to 2200[μs] every 50[μs] interval, total 39 patterns, optimal T_2 and T_3 are decided in the same way. The optimization problem of minimizing the following evaluation function is solved as a decision method.

$$\min_{T_2,T_3} J_c = (\max\theta(t) - 4\alpha)^2, \quad (T_1 + T_2 + T_3 + T_d \leq t) \qquad (12)$$

,where T_d is the settling time of the current for the pulse input. To minimize the overshoot at the desired angle, a Simplex Method is used as an optimization technique. The evaluation function is decided to be the error square of the desired angle and the maximum angle after reaching at around the neighborhood of the desired angle. If the overshoot with the desired angle is small, the vibration then is also a little, because the stepping motor is settled at the desired angle decided according to the excitation phase.

The overshoot of the angle in the control time obtained by a Simplex Method were almost 0.0009[deg] or less. The overshoot was less than 0.1 % against the step angle of $\alpha = 0.9$[deg] by a Simplex Method. Therefore, it is thought that there is little overshoot.

Simulation results using control timing obtained

in a Simplex Method stated above are shown in Figure 7 and Figure 8. Here, Figure 7 shows the time response of rotor angle and angular velocity, and Figure 8 shows the phase plane trajectory. The point in the figure is control time of the pulse input, and the arrows shows transitions of the pulse input timing when T_1 changes from 300[μs] to 2200[μs]. *VelocityErrorPlane* method is commonly used in the phase plane trajectory of the stepping motor to prevent a transverse axis from becoming long (D.Ebihara and T.Iwasa, 1984). Here, in *VelocityErrorPlane* method, the phase difference $\frac{\pi}{2}$ is subtracted whenever the excitation phase is switched. However, in this paper, to see the response to control time of pulse four times, the angle without subtracting the phase difference was drawn in a transverse axis.

There is a width of about 1[ms] to settling to the desired angle using pulse timing in this study. Motion time can be shortened to use the pulse timing in the trajectory with large angular velocity on the phase plane.

When the current dynamics is not considered, the changeover point on the phase plane without vibration by the desired angle is 3.6[deg] and 0[deg/s] (D.Ebihara and T.Iwasa, 1984). However, it is confirmed that the last changeover point is the smaller angle than 3.6 [deg], when there is the transient response of the current. As seen from Figure 7, the time to reach at the desired angle is about 700 [μs] after the fourth input pulse is given, and it corresponds to the transient response of the current for pulse input assumed from Figure3. Consequently, it is confirmed that the excitation sequence with a little vibration can be obtained only by considering the response of the current. Therefore, to realize the vibration-free response in the full step drive, the proposed model considering the current dynamics and the control design based on the model are made clear through these analyses.

5 EXPERIMENTAL RESULTS

Three patterns are selected based on the timing designed in Section 4 and experiments are done. The pulse control time used in the experiments is shown in Table 3. Experimental results of No.1, No.2, and No.3 are shown in Figure 9, Figure 10, and Figure 11, respectively.

Simulation results of both the angles and the angular velocity corresponds well to experimental results. Especially, the vibration of experimental number No.1 was suppressed within 0.01 [deg] as the same as resolution of encoder.

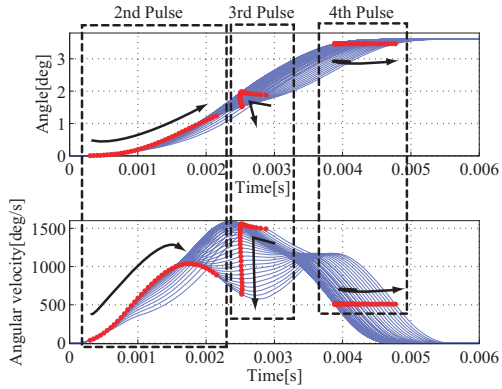


Figure 7: Simulation result of time response using 4 pulse input of various control time.

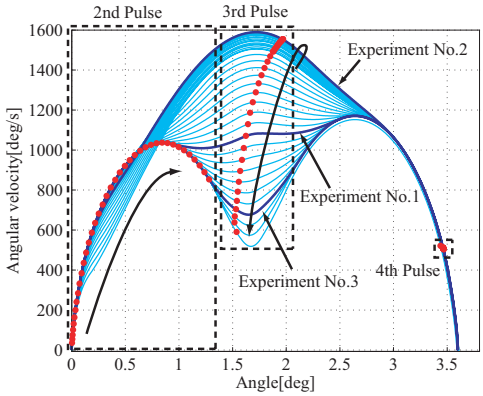


Figure 8: Phase plane trajectory of stepping motor.

Moreover, a vibration of about 810[Hz] is confirmed that is higher than eigenfrequency of the motor in Figure 10 and Figure 11. To realize more highly accurate vibration suppression, it is necessary to consider the vibration in the present model. It is thought that this higher harmonic vibration is due to the influence of the harmonic component of the current, the magnetic field of the drive circuit and the stepping motor.

Through these results, the proposed model considering the response of the current enabled us to achieve the pulse control with little vibration.

Table 3: Pulse timing.

Experiment No.	T_1[μs]	T_2[μs]	T_3[μs]	Total[μs]
No.1	1700	810	1810	4320
No.2	800	1722	1368	3890
No.3	2050	476	2144	4670

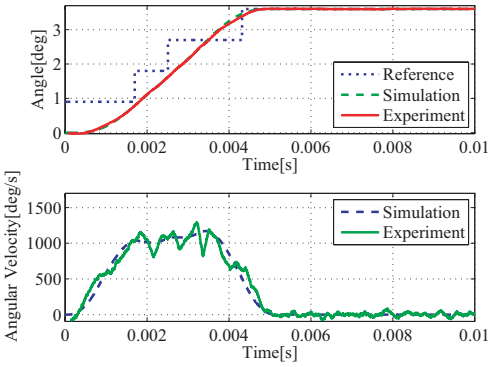


Figure 9: Experimental result of case No.1.

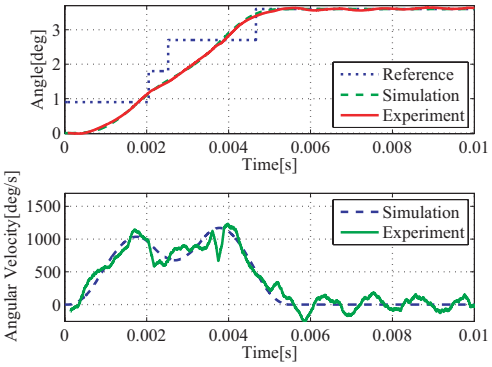


Figure 11: Experimental result of case No.3.

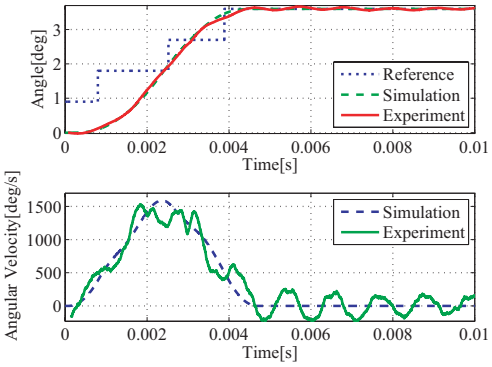


Figure 10: Experimental result of case No.2.

6 CONCLUSIONS

In this paper, a mathematical model by means of the torque equation of the stepping motor considering the dynamics of the current has been built. The relation between the response of the current and the excitation timing on the phase plane were discussed for a full step drive of four pulses. From the simulation, in the system that there is the transient response in the current, it was confirmed that the input timing of the last pulse should be conducted before the end time, considering the duration time of current delay. The validity of the proposed model was shown by the comparison between the simulation and the experiment. The pulse input timing without vibration was decided by means of four pulses, and excellent vibration control was able to be realized based on the proposed model using Simplex Method. The effectiveness of the proposed approach was demonstrated through experiments and simulations.

Robustness to the higher harmonic vibration and model parameter variation by load variation, etc should be investigated, and then robust control must be achieved in near future.

REFERENCES

D.Ebihara and T.Iwasa (1984). *Technology for Use of Stepping Motor in Japanese*. Kogyo Chosakai Publishing Co. Ltd.,, Tokyo.

F.Betin and D.Pinchon (1998). Robust speed control of a stepping motor drive using fuzzy logic. In *Proc IEEE Int Conf Control Appl 1998 Vol.2 pp.948-952*.

K.Laid, D.Xu, and J.Shi (2001). Vector control of hybrid stepping motor position servo system using neural network control. In *Annu Conf IEEE Ind Electron Soc Vol.27 No.2 pp.1504-1508*.

M.Bodson, J.N.Chiasson, R.T.Novotnak, and R.B.Rekowski (1993). High-performance nonlinear feedback control of a permanent magnet stepper motor. In *IEEE Trans Control Syst Technol Vol.1 No.1 pp.5-14*.

M.Hamaguchi, K.Terashima, and H.Nomura (1994). Optimal control of transferring a liquid container for several performance specifications in japanese. In *Trans JSME Vol.60 No.573C pp.1668-1675*.

S.M.Yang and E.L.Kuo (2003). Damping a hybrid stepping motor with estimated position and velocity. In *IEEE Trans Power Electron Vol.18 No.3 pp.880-887*.

T.Miura and T.Taniguchi (1999). Open-loop control of a stepping motor using oscillation-suppressive exciting sequence tuned by genetic algorithm. In *IEEE Trans Ind Electron Vol.46 No.6 pp.1192-1198*.

T.Miura, T.Taniguchi, and H.Dohmeki (2000). Suppression of rotor oscilation in microstepping of a stepping motor by pre-compensator in japanese. In *T.IEE Japan Vol.120-D No.12 pp.1462-1470*.

A JOINT HIERARCHICAL FUZZY-MULTIAGENT MODEL DEALING WITH ROUTE CHOICE PROBLEM
RoSFuzMAS

Habib M. Kammoun, Ilhem Kallel and Adel M. Alimi
Research Group on Intelligent Machines REGIM, University of Sfax, Tunisia
{habib.kammoun, ilhem.kallel, adel.alimi}@ieee.org

Keywords: Transportation Route Choice problem, Intelligent Transportation System, MultiAgent System, Fuzzy Controller.

Abstract: Nowadays, multiagent architectures and traffic simulation agent-based are the most promising strategies for intelligent transportation systems. This paper presents a road supervision model based on fuzzy-multiagent system and simulation, called *RoSFuzMAS*. Thanks to agentification of all components of the transportation system, dynamic agents interact to provide real time information and a preliminary choice of advised routes. To ensure the model rationality, and to improve the route choice make decision, we propose to use a hierarchical Fuzzy inference including some pertinent criteria handling the environment as well as the driver behavior. A multiagent simulator with graphic interface has been achieved to visualize, test and discuss our road supervision system. Experimental results demonstrate the capability of *RoSFuzMAS* to perform a dynamic path choice minimizing traffic jam occurrences by combining multiagent technology and real time fuzzy behaviors.

1 INTRODUCTION

In view of the enormous increase of vehicle number, accidents and traffic jam situations become widespread in all road networks in the world. A solution for these problems is to develop and invest in *Intelligent Transportation System* (*ITS*) which is capable of managing in a better way the existing capacity and encouraging more efficient vehicle routing over time and space, in order to improve safety, traffic efficiency, etc. Varied applications of *ITS* currently under development represent a real opportunity to advance toward a best future.

Furthermore, a number of ITS based on multiagent approach came recently into being to improve performances dynamic routing and traffic management by employing collaborative driving system (Hallé and Chaib-Draa, 2005) or by route guidance system (Adler *et al.* 2005).

Since the nineties, the use of fuzzy logic in *ITS* is marked. Research in soft computing field has been exploring the application of fuzzy set theory as a framework solving many transportation problems (Teodorovic, 1999), as route choice problem, traffic assignment problem, traffic control at the intersection, accident analysis and prevention, and traffic light controller. The majority of authors are based on a comparison of fuzzy values representing the routes' costs. The corresponding rules are of the type: "*If times on route 1 and 2 are very high, I will probably take route 3*".

In this sense, this paper presents a joint hierarchical fuzzy-multiagent model dealing with transportation route choice problem. Our model called *RoSFuzMAS*, acronym for "*Road Supervision based on Fuzzy MultiAgent System*" is poised between two different philosophies: the distributed and parallel *ITS* and the uncertain reasoning. To ensure the model rationality, and to improve the route choice make decision, we propose to use a hierarchical Fuzzy inference including some pertinent criteria handling the environment as well as the driver behavior.

The paper is organized as follows: Next section presents our road supervision system. The third section describes the improvement of decision making for route choice problem by adding other decision criteria structured in a hierarchical fuzzy controller. The simulation part detailed in the forth section gives an idea about the environment and discusses some results.

2 A ROAD SUPERVISION DISTRIBUTED UNDER MULTIAGENT APPROACH

Since some years ago, *multiagent systems* (*MAS*) took hold data processing (Wooldridge, 2002). Indeed, a cooperative interaction always leads to an increase of quantitative and qualitative system performances (Kallel *et al.*, 2002), (Kammoun *et al.*, 2005), (Kallel and Alimi, 2006).

In this sense, our system has as objectives to ensure an efficient network capacity allocation and decrease the number of congestion situations. Accordingly, the system proposes a best road choice to help drivers' vehicle to attempt their destinations.

We propose a model involving three kinds of agents: *City Agent* (*CA*), *Road Supervisor Agent* (*RSA*) and *Intelligent Vehicle Agent* (*IVA*). Figure 1 presents three levels of the proposed system as well as the acquaintance links between *CA*, *RSA* and *IVA*. Each agent use the organizational model *AGRE* (*Agent-Group-Role-Environment*) (Ferber *et al.*, 2005) and lives according to a cycle bound to an iterative process of reception / deliberation / action detailed in (Kallel *et al.*, 2006).

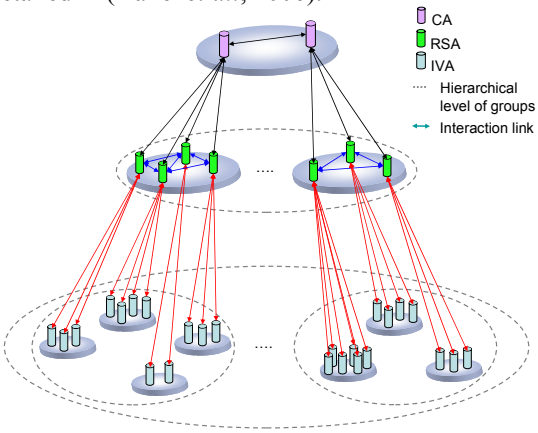


Figure 1: Hierarchical organizational architecture.

The *RSA* computes the traffic index for road *i* (RFI_i) according to equation (1).

$$RFI_i = \frac{N_v}{N_{v\max}} * \frac{\overline{T_j}}{T_t} = \frac{\sum_{j=1}^{N_v} T_j}{N_{v\max} * T_t} \qquad (1)$$

$$\text{with } \overline{T}_j = \text{Average}\,(T_j) = \frac{1}{N_v}\sum_{j=1}^{N_v} T_j$$

with N_v is the number of *vehicles* in road *i*, N_{vmax} is the maximum number of vehicles in this road, T_j represents the time in jam state for vehicle *j* calculating in T_t period.

Equation 2 presents the *Path Flux Index* (*PFI)* as a sum of RFI_i average with the route length pondered by a coefficient α.

$$PFI = \overline{RFI}_i + \alpha \sum_{i=1}^{nb} l_i \qquad (2)$$

with *nb* is the number of road in the path, l_i is the length of road *i* and α is the length importance coefficient.

3 HIERARCHICAL FUZZY ROUTE CHOICE CONTROL

Modelling route choice behaviour is a complex activity if we add other inputs. We try to improve our route choice model by using fuzzy logic (Zadeh, 1965). Furthermore, the use of hierarchical fuzzy controller in several applications' areas showed a real improvement in precision and interpretability (Alimi, 1997), (Kallel *et al.*, 2005) especially in multi-choice problem.

As shown in figure 2, we select only the *k* first paths as *k* alternatives for fuzzy choice, while fuzzifying their *PFI* values. The hierarchical controller uses other inputs fuzzy representations of route characteristic depending on *n* criteria. It provides the recommended route *R* to follow by the vehicle driver.

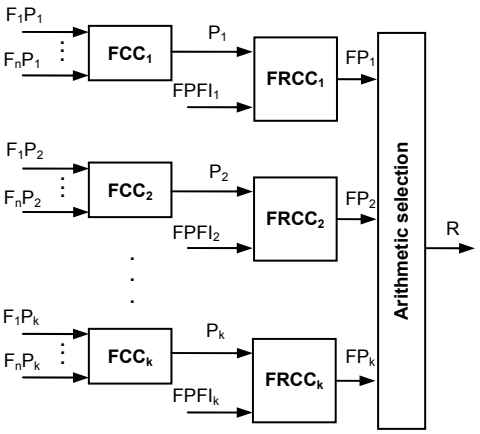


Figure 2: Hierarchical fuzzy route choice model.

with *FiPj* is the fuzzy representation of path *j* corresponding for criteria *i* and R is the recommended route.

3.1 Fuzzy Criteria Controller *FCC*

Let suppose that *k*=3 and *n*=5, we will compare 3 alternative routes depending on 5 factors in urban environment. These factors are the most important

criteria, more used, and accessible from the vehicle information system.

The *FCC* allows a better road evaluation according to criteria concerning the vehicle state, the driver behavior and the environment.

- **Inputs parameters:** these criteria are presented in descending order of their importance for route choice makes decision.
  - RWInformation (road work information, the highest important criteria): NoRoadWork, RoadWork
  - TimeOfDay: Morning, Midday, Evening, Night
  - Familiarity: Unfamiliar (with a route), Medium, Familiar. This parameter takes in account the driver's experiences and will be updated in each trip
  - WeatherConditions: Bad, Medium, Good
  - Speed: Slow, Medium, High
- **Output parameters:**
  - Preference: Weak, Strong

The figure 3 draws the membership function used in this case.

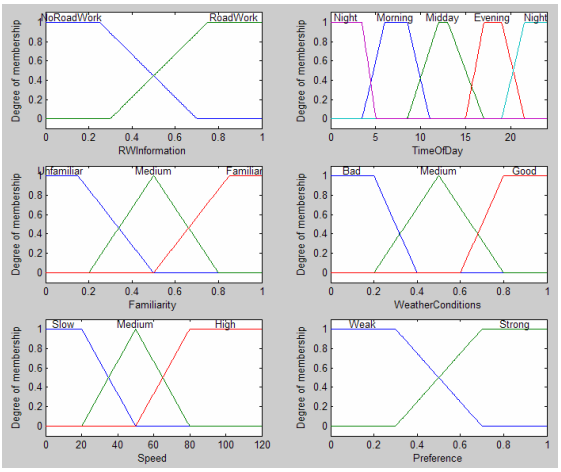


Figure 3: Fuzzification of inputs and output used in FCC.

- **Rule base of *FCC* model:** The rule base of *FCC* model is built by combination of input and output variables. This base is generated by experts in the transportation area and formed initially by 216 rules. As an example of rule, we can cite: "*if RWInformation is NoRoadWork and TimeOfDay is Night and Familiarity is Familiar and WeatherConditions is Good and Speed is Medium then Preference is Strong*".
- **Fuzzy Inference and defuzzification of the *FCC* model:** For the inference process, Mamdani (max–min) inference method is used in FCC model. The Center of Gravity (*COG*) method is used for defuzzification of the *FCC* model.

In view of the fact that the number of rules is high, we propose to model this controller by a hierarchical fuzzy architecture in order to gain in interpretability without decreasing efficiency. We regroup by pairs the criteria having some correlation.

3.2 Fuzzy Route Choice Controller *FRCC*

The *FRCC* uses as inputs, the outputs of *FCC* and a Fuzzy representation the *PFI*, called *FPFI*.

- **Input parameters:**
  - Preference: Weak, Strong
  - FPFI: Low, Middle, High
- **Output parameters:**
  - FP: VeryUnrecommended, Unrecommended, Undecided, Recommended, VeryRecommended
- **Rule base of *FRCC* model:** The rule-base is formed initially by 216 rules. As an example of rule, we can cite: "*if Preference is Strong and FPFI is Low then FP is VeryRecommended*".

4 SIMULATION EXPERIMENTS

Figure 4 presents some virtual maps, created by agent observer of TurtleKit tool (Michel *et al.*, 2005), in order to apply several tests varying vehicles' positions, environment conditions and drivers' behaviour factors.

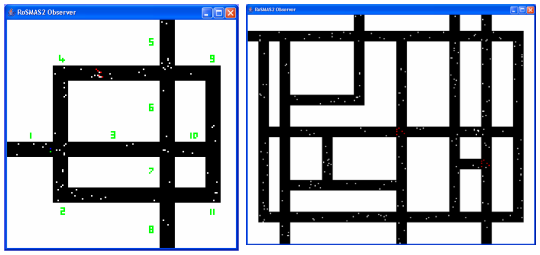


Figure 4: Examples of simulation environments.

The simulator recognizes three kinds of vehicles named *classic vehicle*, *bad vehicle*, and *intelligent vehicle*. The first one is a vehicle without intelligent module; the second one is a vehicle stopped in jam situation; the third one is intelligent, that means it is a part of *RoSFuzMAS*. With several tests, we try to compare intelligent vehicle route choice behaviour with a classic vehicle leaving from the same position in the same time and having the same destination. The first road network presented is a virtual map holding eleven roads numbered from 1 to 11 in only one city, and a variable number of cars circulating with random and autonomous way. The network state in defined time intervals is known as well as

the traffic load intensity to be forwarded from road 1 to road 5. Figure 5 shows the road flux index in the different alternatives from road number 1 to road number 5. The *IVA* chooses the first alternative (by road 4) because it has the smallest flux index compared with second and third alternatives. The flux index in the second alternative is high because of jam situation in road 6. The flux index in the third alternative is high because of the route length. The *RFI* was from 0 to 100 %. The simulation has been done every 450 seconds when updating the road flux index table after every 60 seconds.

Second series of simulations was performed using the fuzzy rule base with the same parameters of the first simulation. Figure 6 confirms that after the work information, bad weather condition, and driver's unfamiliarity of the road 4, the controller proposes the third alternative to follow.

Various other simulations are applied with other maps, other positions of clutters, and different criteria. The results show that the fuzzy logic application for route choosing gives a better management of road network in all cases.

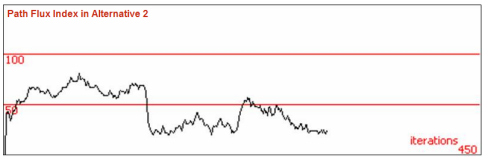


Figure 5: Viewer of Path Traffic Index.

```
[turtle 213]  I'm in road 1. I want to get to Road 5 ?
[IVA 213] 3 possible Paths:
[IVA 213] alternative 1: by Road 4
[IVA 213] alternative 2: by Road 3 + Road 6
[IVA 213] alternative 3: by Road 3 + Road 10 + Road 9
[RSA 1] the flux index in Road 4 is 21 %
[RSA 1] the flux index in Road 3 is 35 %
[RSA 1] the flux index in Road 6 is 73 %
[RSA 1] the flux index in Road 9 is 32 %
[RSA 1] the flux index in Road 10 is 46 %
[IVA 213] alternative 1: RFI = 21 %
[IVA 213] alternative 2: RFI = 54 %
[IVA 213] alternative 3: RFI = 47.66 %
[IVA 213] the recommended path: alternative 1
[IVA 213] the recommended path using Fuzzy controller: alternative 3
[IVA 213] follow alternative 3
[IVA 213] follow road 3
[turtle 213]  to move forward
```

Figure 6: Example of communication messages between IVA and RSA.

5 CONCLUSION

In this paper, we presented a hierarchical architecture as well as a model and a simulation of road supervision system based on joint fuzzy logic and multiagent approach. The route choice algorithm developed shows acceptable results, but it become very complex if we add other criteria for route choice make decision.

The originality of this model resides on:

- A hierarchical fuzzy controller in the multi-route choice problem.
- Generic architecture, without limit for the number of factors to use.
- A hierarchical multiagent architecture handling fuzzy inference for the route choice problem.

As perspectives, we intend in the near future to add other options such as the factor of variant speed for *IAV*, an advancement treatment of crossroads, an environment with double way, and the change lane problem. Applying learning methods such as (Kallel *et al.*, 2006) become a necessity in order to reduce rule numbers and adjust membership functions. Furthermore, paths learning and multiobjective optimization of vehicle path planning can be added.

REFERENCES

Adler, J. L., Satapathy, G., Manikonda, V., Bowles, B., & Blue, V. J., 2005. A multi-agent approach to cooperative traffic management and route guidance. *Transportation Research Part B, 39(4)*, 297-318.

Alimi, A. M., 1997. A neuro-fuzzy approach to recognize Arabic handwritten characters. *In Proc. of the I. C. on Neural Networks, vol. 3*, 1397-1400.

Ferber, J., Michel, F. & Baez, J., 2005. AGRE: Integrating environments with organizations. *In Proc. of E4MAS, Lecture Notes in Artificial Intelligence, 3374*, 48-56.

Hallé, S., Chaib-Draa, B., 2005. A collaborative driving system based on multiagent modelling and simulations. *Transportation Research Part C, 13*, 320-345.

Kallel, I., Jmaiel, M., & Alimi, A. M., 2002. A Multi-Agent Approach for Genetic Algorithm Implementation. *In Proc. of IEEE SMC, Tunisia*.

Kallel, I., Jelleli, T. & Alimi, A. M., 2005. Hierarchical FLS Design Using Multi-Agent Genetic Approach. *In Proc. of GFS'2005*, Granada, Spain, 142-147.

Kallel, I., Alimi, A. M., 2006. MAGAD-BFS: A Learning Method for Beta Fuzzy Systems based on a Multi-Agent Genetic Algorithm. *I. J. of Soft Computing, 10(9)*, 757-772.

Kammoun, H. M., Kallel, I., Alimi, A. M., 2005. RoSMAS²: Road Supervision based Multi-Agent System Simulation. *In Proc. of the I. C. on Machine Intelligence ACIDCA-ICMI'2005*, Tunisia, 203-210.

Michel, F., Beurier, G., & Ferber, J., 2005. The TurtleKit Simulation Platform: Application to Complex Systems. *In Proc. of SITIS'05*, 122-127.

Teodorovic, D., 1999. Fuzzy logic systems for transportation engineering: the state of the art. *Transportation Research Part A, 33(5)*, 337-364.

Wooldridge, M., 2002. *An introduction to multiagent systems*, John Wiley and Sons. February 2002.

Zadeh, L., 1965. Fuzzy sets, *Information and Control 8, 338-353.*

AUTOMATIC ESTIMATION OF PARAMETERS FOR THE HIERARCHICAL REDUCTION OF RULES OF COMPLEX FUZZY CONTROLLERS

Yulia Ledeneva
Instituto Politecnico Nacional, Center of Investigation in Computing, Unidad Adolfo López Mateos,D.F., México
yledeneva@inaoep.mx

Carlos A. Reyes-García, Alejandro Díaz-Méndez
Instituto Nacional de Astrofísica, Óptica y Electrónica, Luis Enrique Erro, Puebla, México
kargaxxi@inaoep.mx, ajdiaz@inaoep.mx

Keywords: Fuzzy control, rule base reduction, hierarchical method, genetic algorithm.

Abstract: The application of fuzzy control to large-scale complex systems is not a trivial task. For such systems the number of the fuzzy IF-THEN rules exponentially explodes. If we have *m* linguistic properties for each of *n* variables, we will have m^n rules combinations of input values. Large-scale systems require special approaches for modelling and control. In our work the system's hierarchical structure is studied in an attempt to reduce the size of the inference engine for large-scale systems. This method reduces the number of rules considerably. But, in order to do so, the adequate parameters should be estimated, which, in the traditional way, depends on the experience and knowledge of a skilled operator. In this work, we are proposing a method to automatically estimate the corresponding parameters for the hierarchical rule base reduction method to be applied to fuzzy control complex systems. In our approach, the parameters of the hierarchical structure are found through the use of genetic algorithms. The implementation process, the simulation experiments and some results are presented.

1 INTRODUCTION

Since the first fuzzy controller was presented by Mamdani in 1974, the different studies devoted to the theory of fuzzy control have shown that the area of development of fuzzy control algorithms has been the most active area of research in the field of fuzzy logic in the last years. From the decade of the 80´s, fuzzy logic has performed a vital function in the advance of practical and simple solutions for a great diversity of applications in engineering and science.

Some fuzzy control applications to industrial processes have produced results superior to its equivalent obtained by classical control systems. The domain of these applications has experienced a serious limitation when expanding it to more complex systems, because a complete theory to determine the performance of the systems when there is a change in its parameters or variables does not yet exist.

When some of these applications are designed for more complex systems, the number of fuzzy rules controlling the process is exponentially increased with the number of variables related to the system. For example, if there are *n* variables and *m* possible linguistic values for each variable, m^n fuzzy rules would be needed to construct a complete fuzzy controller. As *n* grows, the rule base quickly overloads the memory of any computing device, causing difficulties in the implementation and application of the fuzzy controller.

A hierarchical structure is studied in an attempt to reduce the size of the inference engine for large-scale systems. This structure reduces considerably the number of rules. However, the adequate parameters should be estimated. In traditional techniques much reliance has to be put on the experience of the system designer in order to find a good set of parameters (Jamshidi M., 1997).

Genetic algorithms (GA) are an appropriate technique to find parameters in a large search space. They have shown efficient and reliable results in solving optimization problems. For these reasons, in this work we estimate the parameters needed for the rule base reduction method by means of GAs.

The paper is organized as follows. Section 2 summarizes the principles of rule base reduction methods. In Section 3, the hierarchical structure is

described. Section 4 proposes the GA which allows us to automatically find the parameters for the hierarchical structure in order to improve the complex fuzzy control system performance. The results are presented in Section 5.

2 RULE BASE REDUCTION METHODS

The size of the rule base of complex fuzzy control systems grows exponentially with the number of input variables. Due to that fact, the reduction of the rule base is a very important issue for the design of this kind of controllers. Several rule base reduction methods have been developed to reduce the rule base size. For instance, fuzzy clustering is considered to be one of the important techniques for automatic generation of fuzzy rules from numerical examples (Bezdek, 1974). However, for control applications, often there is not enough data for a designer to extract a complete rule base for the controller.

Anwer (Anwer, 2005) proposed a technique for generation and minimization of the number of such rules in case of limited data sets availability. Initial rules for each data pairs are generated and conflicting rules are merged on the basis of their degree of soundness. This technique can be used as an alternative to develop a model when available data may not be sufficient to train the model.

A neuro-fuzzy system (Ajith, 2001; Kasabov, 1998; Chia-Feng, 1998; Jang, 1993; Kim, 1999) is a fuzzy system that uses a learning algorithm derived from, or inspired by, neural network theory to determine its parameters (fuzzy sets and fuzzy rules) by processing data samples. Modern neuro-fuzzy systems are usually represented as special multilayer feedforward neural networks (for example, models like ANFIS (Jang, 1993), HyFis (Kim, 1999), etc.). A disadvantage of these approaches is that the determination of the number of processing nodes, the number of layers, and the interconnections among these nodes and layers are still an art and lack systematic procedures.

Jamshidi (Jamshidi M., 1997) proposed to use sensory fusion to reduce a rule base size. Sensor fusion combines several inputs into one single input. The rule base size is reduced since the number of inputs is reduced. Also, Jamshidi proposed to use the combination of hierarchical and sensory fusion methods. The disadvantage of the design of hierarchical and sensory fused fuzzy controllers is that much reliance has to be put on the experience of the system designer to establish the needed parameters. To solve this problem, we automatically estimate the parameters for the hierarchical method using GAs.

3 HIERARCHICAL METHOD

When a fuzzy controller is designed for a complex system, often several measurable output and actuating input variables are involved. In addition, each variable is represented by a finite number m of linguistic labels. This indicates that the total number of rules is equal to m^n, where n is the number of system variables. As an example, consider $n = 4$ and $m = 5$ than the total number of fuzzy rules will be $k = m^n = 5^4 = 625$. If there were five variables, then we would have $k = 3125$. From the above simple example, it is clear that the application of fuzzy control to any system of significant size would result in a dimensionality explosion.

In the proposed hierarchical fuzzy control structure by Jamshidi (Jamshidi M., 1997), the first-level rules are those related to the most important variables and are gathered to form the first-level hierarchy. The second most important variables, along with the outputs of the first-level, are chosen as inputs to the second-level hierarchy, and so on. Figure 1 shows this hierarchical rule structure. The first and the i-th rule of the hierarchically categorized sets are given by

IF y_1 is A_{1i} and … and y_n is A_{ni} THEN u_1 is B_1

IF y_{Ni+1} is A_{Ni1} and … and y_{Ni+nj} is A_{Ninj} THEN u_i is B_i,

where $i,j = 1, \ldots, n$; y_i are the system's output variables, u_i are the system's control variables, A_{ij} and B_i are linguistic labels; $N_i = \sum_{j=1}^{i-1} n_j \leq n$ and n_j is the number of j-th level system variables used as inputs.

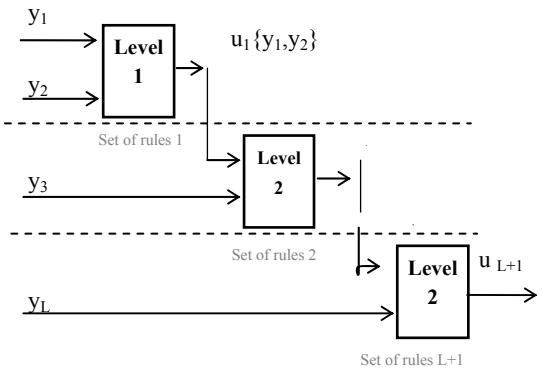


Figure 1: Schematic representation of hierarchical fuzzy controller.

The goal of this hierarchical structure is minimize the number of fuzzy rules from exponential to linear function. Such rule base reduction implies that each system variable provides one parameter to the hierarchical scheme. Currently, the selection of such parameters is manually done and becomes a fastidious and time-consuming routine. In this work, we propose to find these parameters using genetic algorithms.

4 GENETIC ESTIMATION

Nature has an ability to adapt and learn without being told what to do. In other words, genetically nature finds good chromosomes blindly. GAs do the same. Two mechanisms link a GA to the problem it is solving: encoding and evaluation. The GA uses a measure of fitness of individual chromosomes to carry out reproduction. As reproduction takes place, the crossover operator exchanges parts of two single chromosomes, and the mutation operator changes the gene value in some randomly chosen location of the chromosome.

The procedure of estimating the hierarchical variables by GA is summarized as follows:

1. Determine the hierarchical structure and how many parameters we must find.
2. Construct an initial population randomly.
3. Decode each string in the population
4. Evaluate the fitness value for each string.
5. Reproduce strings according to the fitness value calculated in Step 4.
6. Go to 3 until the maximum number of iterations is met.

To start with our algorithm we propose to encode all parameters in one chromosome. For every parameter we will dedicate 8 bits, so we can have the parameters in the range of 2^8 possibilities. All the parameters are positive and have one decimal after the comma, then our range is in [0; 25.6]. The search space can be changed depending on the application. Using this simple encoding procedure we can easily change the number of bits.

Then we evaluate the results using the fitness function which is based on step response specifications such as overshoot, rise time and settling time. We define the fitness function so that it measures how close each individual in the population at time t (i.e., each hierarchical parameter) is to meeting these specification.

Then, after knowing the design specification of the objective function, and once we can obtain the step response characteristics for each chromosome in the population, the fitness function is calculated in 2 steps:

1. We ask if the result coming from the GA is in the range of design specification of the objective function. If it is, we go to the step 2. If it is not, the fitness value of this chromosome is set to 0.
2. The fitness function is defined as

$$FF = (os\_coef - os\_dis)^2 + (ts\_coef - ts\_dis)^2 + (tr\_coef - tr\_dis)^2$$

where *os* is overshoot, *ts* is settling time and *tr* is rising time. The index *coef* is the specification of the control problem for which we are looking the hierarchical parameters. The index *dis* is the design specification parameter. In order to minimize the fitness function we divide 1/FF.

When the evaluation is done, we continue with the reproduction stage. The new population is obtained by applying the crossover operator in one point with probability equal to 0.8 and the mutation operator with probability equal to 0.1.

5 SIMULATION RESULTS

We applied the proposed method in order to find the searching parameters. The proposed method was tested in the inverted pendulum control problem (Nguyen, 2003). This problem consists in the application of a such power to a cart for not allowing a pendulum stem to fall down and to carry the cart to an objective position. The scheme in Figure 2 shows the main components of the system.

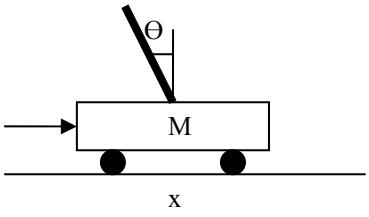


Figure 2: Inverted pendulum.

The basic variables are: angular position of the stem θ, angular velocity of the stem $\Delta\theta$, horizontal position of the cart x, and the velocity of the cart Δx. The design specifications of the inverted pendulum system are: objective position of the cart is 30 cm, overshoot no more than 5%, and settling time no more than 5 sec.

The simulation of the inverted pendulum is performed in *Simulink* of *Matlab* starting from the nonlinear equations (Nguyen, 2003). The fuzzy controllers are implemented in the *Matlab*'s *FIS Editor*. The input fuzzy sets are represented by triangular membership functions (*N*, *Z* and *P*) uniformly distributed in the universe of discourse [-1, 1]. The output fuzzy sets are singletons uniformly distributed in [-1, 1].

The fuzzy controller based on hierarchical structure is composed of three fuzzy controllers (see Figure 3). The total number of rules is of 9 for FC1 + 5 for FC2 + 9 for FC3 = 23 rules.

For applying the reduction with the hierarchical fuzzy controller method we obtained the following parameters: a = 19.2, b = 6.4, c = 1.1, and d= 2.3. With these parameters, the horizontal position of the cart is stabilized in 4.69 seconds with overshoot equal to 0 (see Figure 4), and the behaviour of the angle position of the stem of pendulum is shown in Figure 5.

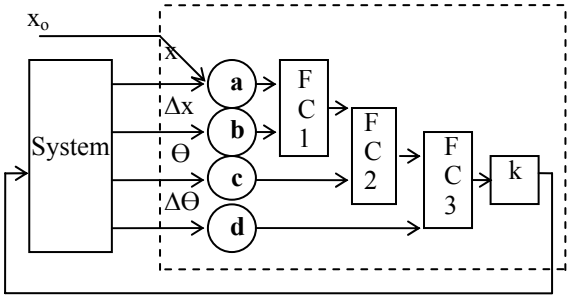


Figure 3: Fuzzy controller based on the hierarchical structure for inverted pendulum.

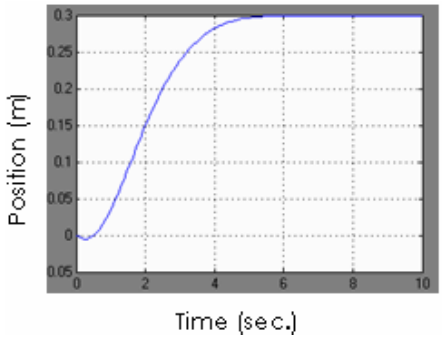


Figure 4: Horizontal position of the cart.

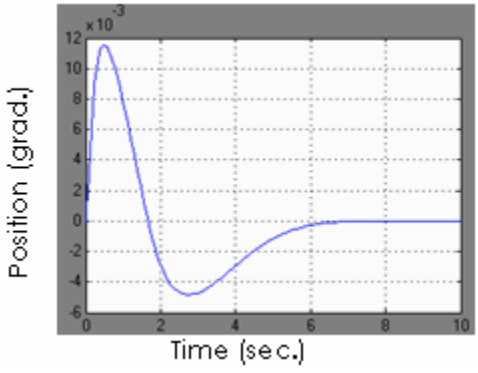


Figure 5: Angle position of the stem of pendulum.

6 CONCLUSIONS

The hierarchical structure makes it possible to significantly reduce the dimensionality of the control problem. In our approach, the problem of manually search for the required parameters was solved with an optimization algorithm (genetic algorithm). The proposed algorithm was tested by simulation of the inverted pendulum control problems. The parameters of the hierarchical method for the design specifications of this problem were adequately found.

Due to the fact that the fitness function is based on the design specification of the system, we have the advantage to apply it to any combination of hierarchical variables. Another very important advantage is that when the user changes the design specifications, we can obtain the necessary hierarchical parameters very quickly by using the proposed GA. GA helped us not only to automatically estimate the hierarchical parameters, but also to improve the results obtained by the hierarchical method.

REFERENCES

Ajith A., 2001. *Neuro-Fyzzy Systems: State-of-the-art. Modeling Techniques. Connectionist Models of Neurons, Learning Processes, and Artificial Intelligence.* Springer-Verlag, LNCS, Jose Mira and Alberto Prieto (Eds.), Spain, vol. 2084, pp. 269-276.

Anwer M.J, 2005. *A Simple Technique for Generation and Minimization of Fuzzy Rules.* IEEE Conference on Fuzzy Systems, Nevada.

Bezdek J.C., 1974. *Cluster validity with fuzzy sets*. J. Cybern., vol. 3, no.3, pp. 58-71.

Chia-Feng Juang, Chin-Teng Lin, 1998. *An On-Line Self-Constructing Neural Fuzzy Inference Network and Its Applications*. IEEE Transaction on Fuzzy Systems, vol. 6, No.1, pp. 12-32.

Jamshidi M., 1997. *Fuzzy Control Systems*. Springer-Verlag, chapter on Soft Computing, pp. 42-56.

Jang Roger, 1993. *ANFIS: Adaptive-Network-Based Fuzzy Inference Systems.* IEEE Trans. System Man & Cybernetics, vol. 23, pp. 665-685.

Kasabov N., Kozma R., and Duch W., 1998. *Rule Extraction from Linguistic Rule Networks and from Fuzzy Neural Networks: Propositional versus Fuzzy Rules.* Proc. of NEURAP'98, France, pp. 403-406.

Kim J., Kasabov N, 1999. *HyFIS: adaptive Neuro-Fuzzy Inference Systems and their application to nonlinear dynamical systems.* Neural Networks, 12, pp. 1301-1319.

Nguyen H.T., Prasad N.R., et al, 2003. *A First Course in Fuzzy and Neural Control.* Chapman & Hall/CRC.

HOLONIC PRODUCTION PROCESS: A MODEL OF COMPLEX, PRECISE, AND GLOBAL SYSTEMS

Edgar Chacon, Isabel Besembel, Dulce Rivero
Universidad de Los Andes, Facultad de Ingeniería, Escuela de Sistemas, Departamento de Computación, Mérida, 5101 Venezuela, email: echacon@ula.ve, ibc@ula.ve, milagro@ula.ve

Juan Cardillo
Universidad de Los Andes, Facultad de Ingeniería, Escuela de Sistemas, Departamento de Sistemas de Control, Mérida, 5101 – Venezuela, email: ijuan@ula.ve

Keywords: Holon, Holonic Production Units, Complex Systems Modelling, Discrete Event Dynamic Systems, Continuous Production Process, Value Chain.

Abstract: Nowadays, it is necessary to have a complete description of the production process in order to plan, program, control, and supervise the production process. It is hard to obtain this description due to the existence of two contradictory points of views. First, the precision implicated in the construction of total and complete models, and on the other hand, the need of having a global vision associated with the different views of the process. These views normally show three important aspects: the structural organization of the model, the dynamism between the main components, and the distinct temporal scales and levels, where are taken the main decisions. The holonic approach (Erikson,2004) has been used to manage this complexity, in order to have an abstraction that permit the integration of the mentioned points of views. In this paper we propose, a structure for continuous production process based on holonic approach in order to obtain a global vision and global model less complex of the production process.

1 INTRODUCTION

Nowadays, enterprises can be established in a virtual manner building a dynamic network of enterprises in order to obtain a determined product, in a moment due. The high enterprises' performance is due to the establishment of precise objectives for a determined configuration of the network.

The virtual enterprise, composed in this way follows a set of production agreements, in order to fulfil objectives by trying to diminish the production cost of the product, in a production offer. This virtual enterprise may be composed by a set of enterprises or a set of unit of the enterprise itself. The production agreements implicate the establishment of a logistic, quantities, and qualities of products and sub-products, and synchronization points (E. Chacon,1998), (E. Chacon, 2004), (Juan Cardillo, 2005).

The negotiation is obtained following the existing common knowledge, for which it is necessary to have the following:

- A description language of the production methods
- Protocols for the acceptation of missions among the enterprise participants, following each one of production capacities under a production method of high level.

The enterprises participating in a negotiation know the information services that offer each enterprise by means of a yellow pages service. The global model of a virtual enterprise focused in its mission, can be viewed as an Holon. The knowledge model of the holon follows a Discrete Event Dynamic Systems Model (DEDS), where each one of the steps that conforms the mission is described by means of an operation region. Thus, a conformed enterprise (virtual or not) is considered as an enterprise holon which is composed by a set of Production Units (inside production units or enterprises) conforming the Production Units Holonic System. The coordinator of the conformed enterprise is charged to manage, control, and re-plan each one of the steps of the mission in order to

complete it. The established chronogram of the conformed enterprise generates a particular mission for each one of the Production Unit Holon (PUH). Each step of the selected production method by the PUH is associated to an operation region. The coordinator of each PUH is charged to manage, control, and re-plan each one of the steps of the selected production method. In holonic architecture, the groups are managed by itself following its internal resources state knowledge, the production order advance, and the knowing of its production method that permits to obtain the product. Thus, a holon is composed by one or more holons. The advantage of use a holonic approach is due to have a reference model that describes the composition of the conformed enterprise (virtual or not) structure. Here, a conformed enterprise and a holon are modeled by following a business model where the value chain and the product flow both establish the base of the global modelling of a conformed enterprise. In this work, we present a reference model under the holonic approach that permits to have a description of the production process (conformed enterprise) as an embedded system based on its business model, value chain, and product flow. This presents one enterprise (conformed enterprise) as a network of enterprises (production units) composed to follow a production mission. Section 2 is devoted to show the conformed enterprise modelling method. Section 3 describes the holonic approach of the production system, and section 4 presents the conclusions and future works.

2 MODELING OF A CONFORMED ENTERPRISE

A conformed enterprise describes both enterprises composed by several semi-independent units or virtual enterprises. This is due to the utilization of a production model in conjunction with a value chain and a production flow. Both enterprises are modelled in the same manner. Thus, the value chain expresses the sequence of the aggregate value of a product (transformation, storage or transport) by following of the production process itself. The use of the value chains is the base to develop models of the different business process that are specific of an enterprise. A graphics representation of the value chains is shown in figure 1. The product flow can be defined as the different transformation stages, which follows a resource (or a set of them) until the final product achieving. The conjunction of the value chain plus the product flow results the production flow, which is the aggregate of functionalities and transformations of the resources to generate the final product.

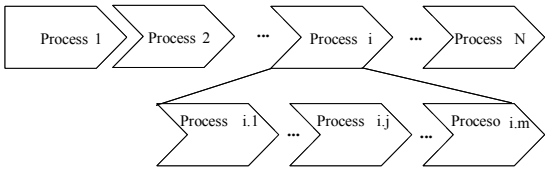


Figure 1: Value chains.

Each stage of the value chain (Input resources, processing or transformation and storage) of the production flow is also viewed as a Production Unit (PU), where the characterization of each PU depends on how the resource (or a set of them) evolves, such as: continuous, batch, manufacture, or hybrid. Additionally, each PU does a specific transformation depending on the properties of the resource (or resources). However, it is possible to found common or generic elements that characterize a PU in a production flow. Each element is viewed as a process inside of the PU. These processes are as follows:

- A process to take hold of resources
- A process of transformation/transport
- A process of storage between each PU process

Initially, resources are located and obtained for the PU. The process to take hold of PU resources warrants resources for a determined production recipe. After, the PU selects a production method required to transform the raw material. The selection of the production method depends on the resources properties, and then it is carrying out the transformation process. When the transformation process is finished, the transformed resource is storing and waiting for the need of another PU.

Figure 2 shows a structural model of a PU. This model not only presents the controlled and supervised system (control and supervision process plus the production process) that makes the transformation, also beholds the product plan, production methods, configuration, and management of the resources that are needed in the production process. This structural description is the base to extract the main information to do a planning of the PU by considering which values or variables permit the description of the PU state, such as: performance indicators, reliability, and so on. Other variables that may be taken into account are: quality of the product, expected quantity of product, production capacity, storage capacity (minimal and maximal), etc. For example, PU production capacity is related to the transformation process capacity, and also to

the storage capacity, if and only if raw material and the rest of the resources are guaranteed.

An object-oriented structural model of the PU drawing by means of the Unified modeling Language (UML) (JçI. Jacobson, www.rational.com/uml), (A. Muller, 1997) is presented in figure 3. The class diagram uses rectangles to represent classes and lines to represent relationships among classes. This diagram includes three kinds of relationships, such as: generalizations/specializations (arrows not fulfilled), associations (lines), and compositions (line finished in a filled diamond).

Figure 3 shows different entities or classes in the PUH and how they are related. In particular, it is highlighted a special class which is related to an association between the classes ProductionPlan and Product, named ProductPlan. Also, the class Configuration which is charged to register all of the different configurations of resources, production process, control and supervision software, and production method, as an association class between the four aforementioned entities. It is viewed the classification of the resources managed by the PU in order to accomplish its production plan that support the enterprise plan.

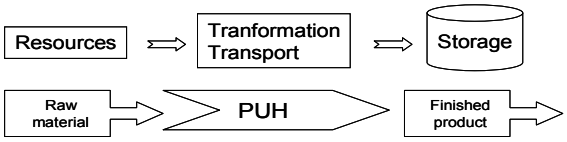


Figure 2: Model of a Production Unit.

For lack of space, this class diagram is showed in a concise form; each class has a set of properties (attributes and relationships) and operations, which support the behavioural model of the PU.

The embedded model of figure 4 shows the decisions scheme needed for each path of the value chain of a PU. Our proposition uses an embedded model that is based on the description of a Holon.

This model presents level works composed by the production process and each one of the established control loop. The designed controllers (that takes decisions) are inside the communications and information architecture, which needs applications and industrial networks in order to capture actual variable values, by means of sensors, and to indicate controller actions, by means of actuators. Thus, the controlled productive process is viewed as a system that need to be controlled (supervise, monitor, manage) by a supervisor. This permits to view the whole productive system as an embedded system conformed by a path of the value chain that takes products, transform-them to generate another derived product.

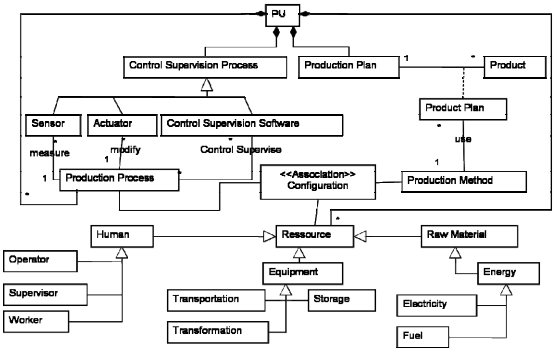


Figure 3: UML Class Diagram of a Production Unit.

3 HOLONIC APPROACH IN PRODUCTION SYSTEMS

A holon for a manufacturing enterprise is defined as a constructor block; cooperative and autonomous for transforming, transporting, storing and/or validating physical objects and information (H. Brusel, 1998). A holon has the autonomy to create and control the execution of its owns plans, it may cooperate with other holons to jointly develop an acceptable plan to reach the system mission. The cooperation among holons is accomplished by one evolution of the holarchy in the organization (a holon system).

In a Holonic system production, the objective is to achieve a complete spectrum of the range of the control function that goes from the production plans, that controller at the highest level, until the process/machine that control the lowest level.

A Holon possesses two constituent elements: a connected transformation system and a system of taking of decisions. The decisions system monitors the resources and the evolution of the order that are being controlled in the plant floor, by associate elements for such an end.

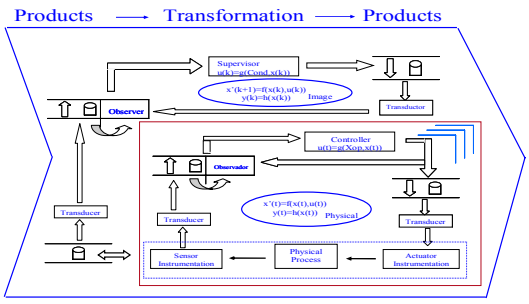


Figure 4: Production process embedding model.

The integral automation of systems outlines a global vision of the productive process, where each element that intervenes in the production should be taken into account to be able ***to control***, **to supervise**, and **to management** the production, to see figure 5. The automation scheme is based on the construction of models that represent the ***Unit of Production*** so much in its structural aspect, like in its dynamics. The control schemes assure that the behavior of the system is inside that wanted, for that which the knowledge that one has of the state of the system allows to evaluate which should be the viable control actions with the purpose of assuring that the system reaches the wanted state.

In the productive system, the control is subordinated to the objective fixed to the **Unit of Production**, and the objectives are determined for what the Unit of Production can make.

The process has a proper behaviour that depends on the physical and chemical laws in particular in a condition of given operation. This behaviour it can describe formally as a Hybrid Dynamic System.

When having the description of the dynamics, the behaviour can be controlled by a automatic supervisory system or by means of the intervention of a human being.

The product of the production process is obtained by means of a group of resources that can suffer changes in the time, thus the information of the state of the process, and of the resources should be continually monitoring with the purpose of determine, if the system completed the production objective, or on the contrary it could not complete the production objective by failure in some of the resources. This change should be management to define a new production scheme in the Unit of Production or to verify in another level if it is possible to have a form of assuring the objective fixed for the set of Units of Production.

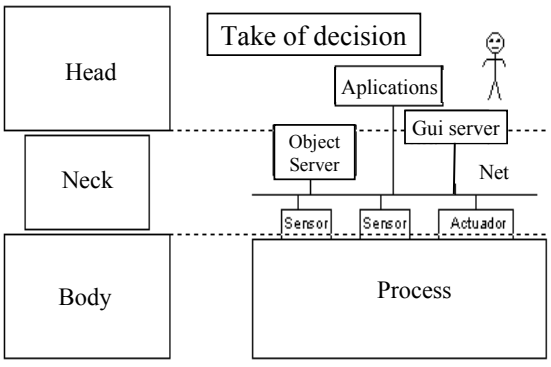


Figure 5: Integral automation of systems.

Considering the Unit of Production with the capacity to have autonomy (a Holon: Holon Unit of Production, HUP). HUP is composing of a BODY, NECK, and HEAD. In the body is where the processes: of transformation of the matter, of storage or transport are developed. This it is carried out by a group of physical devices as reactors, compressors, store, etc.. In the head are the mechanisms of taking of decisions, based on the knowledge of the production process and the necessary resources. These mechanisms of taking of decisions are developed by the classic techniques of supervisory control or by approach of intelligent systems. The neck is the interface between both, this is composed by the whole infrastructure tele-informatics that stores and transports the information. Thus, in the head are the applications of taking of decisions that determine the behaviour that should have the body. In the neck is the communication mechanism between the head and the body (process), their implementation is all the mechanisms that allow to capture information of the process and to send commands to the process. See figure 6.

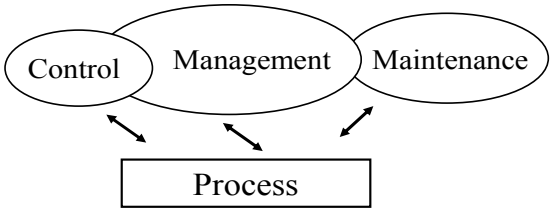


Figure 6: Relation ships between Holon and TIC infrastructure.

3.1 Holonic Control Loop

The basic functional unit for the automation of a production system is the control loop. This control loop is redefined how the Holonic Control Loop (HCL), it possesses all the characteristics of a Holon.

The HLC is conformed by a body that contains the physical process that possesses implicit the actuator and the set of sensors and whose model, without losing generality, we can describe as a dynamic system in state equations. The neck of the HCL this conformed by the architecture tele-informatics and proper applications that are able to capture, to try, to store, to adapt, and to transfer so much information of the sensor as toward the actuator. The head of the HCL (controller) is conformed by the mechanism of taking of decisions, this mechanism can have a rigid form (PID, etc.) or not (neuronal net, etc.). This controller, that is able to regulate to an operation

point, is designed using the physical model of the process (knowledge model), its can be described by dynamic equations and /or algebraic that dependent of the operation point, of the state or output and of parameters that belong to an operation region. In definitive the Holonic Control Loop is an autonomous system that possesses two input and two outputs characterized by products and information. Input Product: products supplies and resource. Input Information: Controller's type, Operation Point, Parameter of Controller. Output Product: sub-product or finished product. Output Information: State of the controller process. To see figure 7.

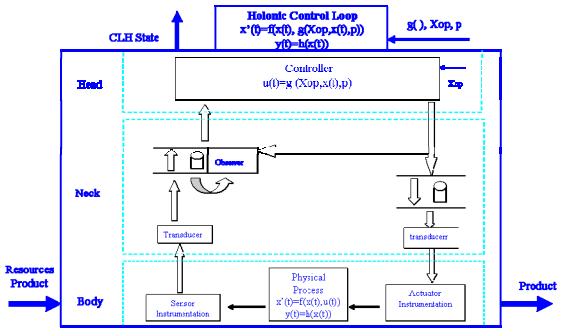


Figure 7: Holonic Control Loop.

3.2 Supervisor Holon

As we know, a process possesses more than a control loop. To each control loop we have redefined as the HCL. The management of all the loop control relapses in a supervisor. The supervisor possesses all the characteristics of a holon. Thus, when composing all the loops control we obtain the Controlled Holonic Systems. See figure 8.

The Controlled Holonic System is a system that has i models (maybe one for each control loop), each model has a set the m of nominal values of operation that can be reached with a set the j types of controllers which are adjusted under an approach determined with n parameters. Just as it shows it the figure 9

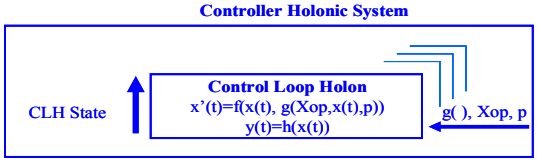


Figure 8: Controller Holonic System.

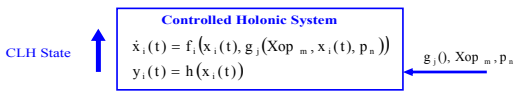


Figure 9: Embedding Controlled Holonic System.

In the same manner that the previous sub-section the Supervisor Holon (SH) is conformed by a body, a neck and head. The Body of SH is conformed by the Controlled Holonic System which generates state values of the process (as sensors), and possesses implicit actuator given by: g (), Xop, p. The neck of SH is conformed by the tele-informatics architecture and set of applications that allow to detect events (they are able to capture, to try, to store, to adapt, continuous information of the state variable in events) as generating the set point (g (), Xop, p) toward the loops control. The head of the SH this conformed by the mechanism of taking of decisions, that is design using approach of modelling of Discrete Event Systems generated by the Controlled Holonic System.

In definitive the Supervisor Holon is an autonomous system that possesses two input and two outputs characterized by products and information. Input Product: products supplies and resource. Input Information: recipe, scheduling of Operation. Output Product: sub-product or finished product. Output Information: State of the supervisor process. See figure 10.

3.3 Holonic Production Unit

It is possible that a production process has more than a supervisor. To each supervisor we have redefined it as the Supervisor's Holon. The management of all the supervisors relapses in a coordinator; this coordinator possesses all the characteristics of a holon: Production Unit Holon. Thus, when we compose all the supervisors, we obtain the Supervised Holonic System. (See figure 11).

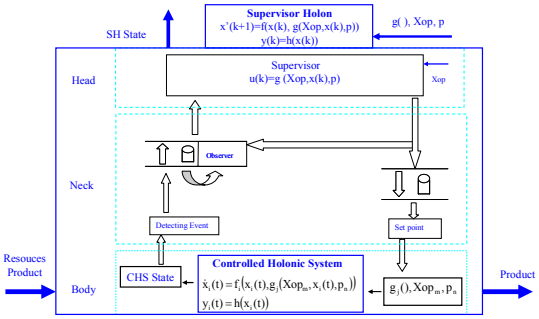


Figure 10: Supervisor Holon.

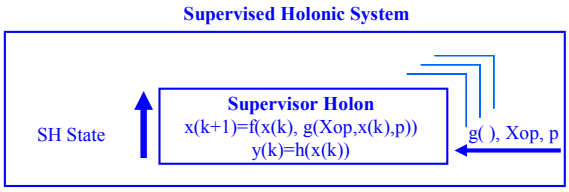


Figure 11: Supervised Holonic System.

The Supervised Holonic System is a system that has i models (maybe but of one for supervisor), each model has a set m of nominal values of operation that can be reached with a combined j of types of recipes which are adjusted under an approach determined by supervisor with n of parameters. Just as is shown it the figure 12.

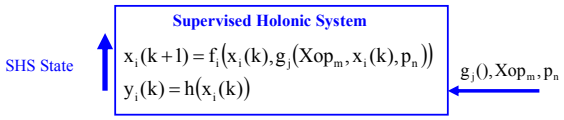


Figure 12: Embedding Supervised Holonic System.

In the same manner that the previous sub-section the Production Unit Holon (PUH) is conformed by a body, a neck and head. The Body of PUH is conformed by the Supervised Holonic System which generates state values of the supervisor process (as sensors), and possesses implicit actuator given by: g (), Xop, p., i.e. recipes, scheduling, etc.. The neck of PUH is conformed by the tele-informatics architecture and set of applications that allow to detect events (they are able to capture, to try, to store, to adapt, information of the state variable of the supervised process in events) as generating the set point (g (), Xop, p) toward the supervisor. The head of the PUH this conformed by the mechanism of taking of decisions, that is design using approach of modeling of Discrete Event Systems generated by the Supervised Holonic System.

In definitive the Production Unit Holon is an autonomous system that possesses two input and two output characterized by products and information. Input Product: products supplies and resource. Input Information: negotiated production demand. Output Product: finished product. Output Information: State of the productive process. See figure 13.

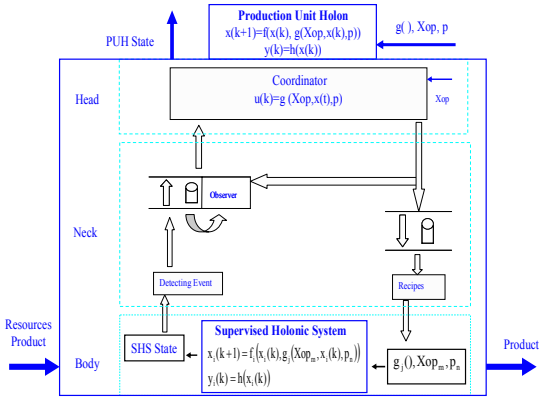


Figure 13: Production Unit Holonic.

4 CONCLUSION AND FUTURE WORK

In the work, we show an implemented architecture that allows to have a recursive (embedded) structure, represented by each Holon. The information stays up-to-date by means of mechanisms topologically equal. The coordination is carried out by means of supervisors generated for each built configuration based on a production mission. The stages for the supervisor's construction are defined like one of negotiation. This negotiation to define the objective and the generation of synchronization points based on the defined configuration. Once finished the stages of selection of the configuration and establishment of the synchronization points the supervisor you instance for the duration of the mission.

It is necessary to establish negotiation mechanisms for the establishment of the mission, since the global knowledge of the production capacities is known internally, and the configuration is obtained after the evaluation of the different arisen alternatives of each participant's internal capacities. The protocol Contract Net of the FIPA appears as the most suitable mechanism for the establishment of a common mission.

ACKNOWLEDGEMENTS

This work is partially supported by the ECOS-NORD Program France-Venezuela on virtual net of production

REFERENCES

H. Brusel, J. Wyns, P. Valckenaers, L. Bongaerts y P. Peeters. Reference architecture for holonic manufacturing systems: PROSA. *Computer in Industry*. Vol. 37. 1998. pp. 255-274.

E. Chacón. *Automatización Integral de Sistemas de Producción Continuos: Tecnologías de Integración y Automatización*. Reporte técnico de la Universidad de Los Andes. Mérida. Venezuela. Diciembre, 1998.

E. Chacón, I. Besembel, F. Narciso, J. Montilva y E. Colina. An Integration Architecture for the Automation of Continuous Production Complexes. *ISA Transactions. Journal of the Americna Intitute of Physics*. Vol 41, Nº 1, pag. 95-113,2002.

E. Chacón, I. Besembel, J-C Hennet. Coordination and Optimization in Oil and Gas Production complexes. Computer in Industry. Elsevier Vol. 53, pag. 17-37, 2004.

G. Doumeingts, B. Vallespir, M. Zanettin y D. Chen. Cim grai integrated methodology, a methodology for designing cim systems. Reporte técnico de la Universidad de Bordeaux, Francia. Mayo. 1992.

International Product Data Management Users Group. Integrating/Interfacing PDM (Product Data Management) with MRP II (Manufacturing Resource Planning) http://www.pdmic.com/IPDMUG/ wpipdmug.html - model. 1996.

I. Jacobson, G. Booch y J. Rumbaugh. Specification of the UML. Rational Software . http://www.rational.com/uml/

K. Kosanke, F. Vernadat y M. Zelm. CIMOSA: Enterprise engineering and integration. *Computer in Industry*. Vol. 40. Elsevier Science. 1999. pp. 83-97.

J. Montilva, E. Chacón y E. Colina. METAS: Un método para la automatización integrada en sistemas de producción continua. *Actas de las IV Jornadas Panamericanas de Automatización*. Caracas. Mayo 2000.

A. Muller. *Modelado de Objetos con UML*. Eyrolles y Ediciones Gestión 2000, S. A., 1997.

PERA. Pera reference model for CIM. ISA publication. http://www.pera.net

PRM. Production Resource Manager. IBM. Last Update april 1996. http://www.research.ibm.com/ pdtr/prm.html

Ramadge, P .J. G. and Wonham, W. M. (1989) The control of discrete event systems. Proceedings of the IEEE, 77 (1): 81-87.

T. J. Williams. A Reference Model for Computer Integrated Manufacturing (CIM). International Purdue Works.

Juan Cardillo, Edgar Chacón, Isabel Besembel, Milagros Rivero, Unidad de Producción como célula fundamental de los procesos Holónicos de Producción, V Congreso de Automatización y Control, CAC 2005, Universidad Simón Bolívar, Caracas-Venezuela

Eriksson, Penker, Lyons, Fado. UML 2 Toolkit. Wiley, 2004.

PATTERN-DRIVEN REUSE OF EMBEDDED CONTROL DESIGN
Behavioral and Architectural Specifications in Embedded Control System Designs

Miroslav Sveda, Ondrej Rysavy
Faculty of Information Technology, Brno University of Technology, Bozetechova 2, 61266 Brno, Czech Republic
sveda@fit.vutbr.cz

Radimir Vrba
Faculty of Electrical Engineering & Communication, Brno University of Technology, Brno, Czech Republic
vrbar@feec.vutbr.cz

Keywords: Embedded systems, Formal specification, Finite automata, Timed automata, Case-based reasoning.

Abstract: This paper deals with reuse of architectural and behavioral specifications of embedded systems employing finite-state and timed automata. The contribution proposes not only how to represent a system's formal specification as an application pattern structure of specification fragments, but also how to measure similarity of formal specifications for retrieval with case-based reasoning support. The paper provides also an insight into case-based reasoning support as applied to formal specification reuse by application patterns built on finite-state and timed automata. Those application patterns create a base for a pattern language supporting reuse-oriented design process for a class of real-time embedded systems.

1 INTRODUCTION

Methods and approaches in systems engineering are often based on the results of empirical observations or on individual success stories. Every real-world embedded system design stems from decisions based on an application domain knowledge that includes facts about some previous design practice. Evidently, such decisions relate to system architecture components, called in this paper as application patterns, which determine not only a required system behavior but also some presupposed implementation principles. Application patterns should respect those particular solutions that were successful in previous relevant design cases. While focused on the system architecture range that covers more than software components, the application patterns look in many features like well-known software object-oriented design concepts such as reusable patterns (Coad and Yourdon, 1990), design patterns (Gamma et. al., 1995), and frameworks (Johnson, 1997). By the way, there are also other related concepts such as use cases (Jacobson, 1992), architectural styles (Shaw and Garlan, 1996), or templates (Turner, 1997), which could be utilized for the purpose of this paper instead of introducing a novel notion. Nevertheless, application pattern can structure behavioral specifications and, concurrently, they can support architectural components specification reuse.

Nowadays, industrial scale reusability frequently requires a knowledge-based support. Case-based reasoning (see e.g. Kolodner, 1993) can provide such a support. The method differs from other rather traditional procedures of Artificial Intelligence relying on case history: for a new problem, it strives for a similar old solution saved in a case library. Any case library serves as a knowledge base of a case-based reasoning system. The system acquires knowledge from old cases while learning can be achieved accumulating new cases. Solving a new case, the most similar old case is retrieved from the case library. The suggested solution of a new case is generated in conformity with the retrieved old case.

This paper proposes not only how to represent a system's formal specification as an application pattern structure of specification fragments, but also how to measure similarity of formal specifications for retrieval. In this paper, case-based reasoning support to reuse is focused on specifications by finite-state and timed automata, or by state and

timed-state sequences. The same principles can be applied for specifications by temporal and real-time logics.

The following sections of this paper introduce the principles of design reuse applied by the way of application patterns. Then, employing application patterns fitting a class of real-time embedded systems, the kernel of this contribution presents two design projects: petrol pumping station dispenser controller and multiple lift control system. Via identification of the identical or similar application patterns in both design cases, this contribution proves the possibility to reuse substantial parts of formal specifications in a relevant sub-domain of embedded systems. The last part of the paper deals with knowledge-based support for this reuse process applying case-based reasoning paradigm. The paper provides principles of case-based reasoning support to reuse in frame of formal specification-based system design aiming at industrial applications domain.

2 STATE OF THE ART

To reuse an application pattern, whose implementation usually consists both of software and hardware components, it means to reuse its formal specification, development of which is very expensive and, consequently, worthwhile for reuse. This paper is aimed at behavioral specifications employing state or timed-state sequences, which correspond to Kripke style semantics of linear, discrete time temporal or real-time logics, and at their closed-form descriptions by finite-state or timed automata (Alur and Henzinger, 1992). Geppert and Roessler (2001) present a reuse-driven SDL design methodology that appears closely related approach to the problem discussed in this contribution.

Software design reuse belongs to highly published topics for almost 20 years, see namely Frakes and Kang (2005), but also Arora and Kulkarni (1998), Sutcliffe and Maiden (1998), Mili et al. (1997), Holzblatt et al. (1997), and Henninger (1997). Namely the state-dependent specification-based approach discussed by Zaremski et. al. (1997) and by van Lamsweerde and Wilmet (1998) inspired the application patterns handling presented in the current paper. To relate application patterns to the previously mentioned software oriented concepts more definitely, the inherited characteristics of the archetypal terminology, omitting namely their exclusive software orientation, can be restated as follows. A pattern describes a problem to be solved, a solution, and the context in which that solution works. Patterns are supposed to describe recurring solutions that have stood the test of time. Design patterns are the micro-architectural elements of frameworks. A framework -- which represents a generic application that allows creating different applications from an application sub-domain -- is an integrated set of patterns that can be reused. While each pattern describes a decision point in the development of an application, a pattern language is the organized collection of patterns for a particular application domain, and becomes an auxiliary method that guides the development process, see the pioneer work by Alexander (1977).

Application patterns correspond not only to design patterns but also to frameworks while respecting multi-layer hierarchical structures. Embodying domain knowledge, application patterns deal both with requirement and implementation specifications (Shaw and Garlan, 1996). In fact, a precise characterization of the way, in which implementation specifications and requirements differ, depends on the precise location of the interface between an embedded system, which is to be implemented, and its environment, which generates requirements on system's services. However, there are no strict boundaries in between: both implementation specifications and requirements rely on designer's view, i.e. also on application patterns employed.

A design reuse process involves several necessary reuse tasks that can be grouped into two categories: supply-side and demand-side reuse (Sen, 1997). Supply-side reuse tasks include identification, creation, and classification of reusable artifacts. Demand-side reuse tasks include namely retrieval, adaptation, and storage of reusable artifacts. For the purpose of this paper, the reusable artifacts are represented by application patterns.

The following two sections provide two case studies, based on implemented design projects, using application patterns that enable to discuss concrete examples of application patterns reusability.

3 PETROL DISPENSER CONTROL SYSTEM

The first case study pertains to a petrol pumping station dispenser with a distributed, multiple microcomputer counter/controller (for more details see Sveda, 1996). A dispenser controller is

interconnected with its environment through an interface with volume meter (input), pump motor (output), main and by-pass valves (outputs) that enable full or throttled flow, release signal (input) generated by cashier, unhooked nozzle detection (input), product's unit price (input), and volume and price displays (outputs).

3.1 Two-level Structure for Dispenser Control

The first employed application pattern stems from the two-level structure proposed by Xinyao et al. (1994): the higher level behaves as an event-driven component, and the lower level behaves as a set of real-time interconnected components. The behavior of the higher level component can be described by the following state sequences of a finite-state automaton with states "blocked-idle," "ready," "full fuel," "throttled" and "closed," and with inputs "release," (nozzle) "hung on/off," "close" (the preset or maximal displayable volume achieved), "throttle" (to slow down the flow to enable exact dosage) and "error":

blocked-idle $\xrightarrow{release}$ ready $\xrightarrow{hung\ off}$ full_fuel $\xrightarrow{hung\ on}$ blocked-idle
blocked-idle $\xrightarrow{release}$ ready $\xrightarrow{hung\ off}$ full_fuel $\xrightarrow{throttle}$ throttled $\xrightarrow{hung\ on}$ $\xrightarrow{hung\ on}$ blocked-idle
blocked-idle $\xrightarrow{release}$ ready $\xrightarrow{hung\ off}$ full_fuel $\xrightarrow{throttle}$ throttled $\xrightarrow{close}$ $\xrightarrow{close}$ closed $\xrightarrow{hung\ on}$ blocked-idle
blocked-idle $\xrightarrow{error}$ blocked-error
blocked-idle $\xrightarrow{release}$ ready $\xrightarrow{error}$ blocked-error
blocked-idle $\xrightarrow{release}$ ready $\xrightarrow{hung\ off}$ full_fuel $\xrightarrow{error}$ blocked-error
blocked-idle $\xrightarrow{release}$ ready $\xrightarrow{hung\ off}$ full_fuel $\xrightarrow{throttle}$ throttled $\xrightarrow{error}$ $\xrightarrow{error}$ blocked-error

The states "full_fuel" and "throttled" appear to be hazardous from the viewpoint of unchecked flow because the motor is on and the liquid is under pressure -- the only nozzle valve controls an issue in this case. Also, the state "ready" tends to be hazardous: when the nozzle is unhooked, the system transfers to the state "full_fuel" with flow enabled. Hence, the accepted fail-stop conception necessitates the detected error management in the form of transition to the state "blocked-error." To initiate such a transition for flow blocking, the error detection in the hazardous states is necessary. On the other hand, the state "blocked-idle" is safe because the input signal "release" can be masked out by the system that, when some failure is detected, performs the internal transition from "blocked-idle" to "blocked-error."

3.2 Incremental Measurement for Flow Control

The volume measurement and flow control represent the main functions of the hazardous states. The next applied application pattern, incremental measurement, means the recognition and counting of elementary volumes represented by rectangular impulses, which are generated by a photoelectric pulse generator. The maximal frequency of impulses and a pattern for their recognition depend on electro-magnetic interference characteristics. The lower-level application patterns are in this case a noise-tolerant impulse detector and a checking reversible counter. The first one represents a clock-timed impulse-recognition automaton that implements the periodic sampling of its input with values 0 and 1. This automaton with b states recognizes an impulse after b/2 (b>=4) samples with the value 1 followed by b/2 samples with the value 0, possibly interleaved by induced error values, see an example timed-state sequence:

$(0, q_1) \xrightarrow{inp=0} \ldots \xrightarrow{inp=0} (i, q_1) \xrightarrow{inp=1} (i+1, q_2) \xrightarrow{inp=0} \ldots \xrightarrow{inp=0} (j, q_2) \ldots$
$\ldots \xrightarrow{inp=1} (k, q_{b/2+1}) \xrightarrow{inp=1} \ldots$
$\ldots \xrightarrow{inp=1} (m, q_{b-1}) \xrightarrow{inp=0} (m+1, q_b) \xrightarrow{inp=1} \ldots \xrightarrow{inp=1} (n, q_b) \xrightarrow{inp=0/IMP} (n+1, q_1)$
i, j, k, m, n are integers representing discrete time instances in increasing order.

For the sake of fault-detection requirements, the incremental detector and transfer path are doubled. Consequently, the second, identical noise-tolerant impulse detector appears necessary.

The subsequent lower-level application pattern used provides a checking reversible counter, which starts with the value (h + l)/2 and increments or decrements that value according to the "impulse detected" outputs from the first or the second recognition automaton. Overflow or underflow of the pre-set values of h or l indicates an error. Another counter that counts the recognized impulses from one of the recognition automata maintains the whole measured volume. The output of the letter automaton refines to two displays with local memories not only for the reason of robustness (they can be compared) but also for functional requirements (double-face stand). To guarantee the overall fault detection capability of the device, it is necessary also to consider checking the counter. This task can be maintained by an I/O watchdog application pattern that can compare input impulses from the photoelectric pulse generator and the changes of the total value; evidently, the appropriate automaton provides again reversible counting.

3.3 Fault Maintenance Concepts

The methods used accomplish the fault management in the form of (a) hazardous state reachability control and (b) hazardous state maintenance. In safe states, the lift cabins are fixed at any floors. The system is allowed to reach any hazardous state when all relevant processors successfully passed the start-up checks of inputs and monitored outputs and of appropriate communication status. The hazardous state maintenance includes operational checks and, for shaft controller, the fail-stop support by two watchdog processors performing consistency checking for both execution processors. To comply with safety-critical conception, all critical inputs and monitored outputs are doubled and compared; when the relevant signals differ, the respective lift is either forced (in case of need with the help of an substitute drive if the shaft controller is disconnected) to reach the nearest floor and to stay blocked, or (in the case of maintenance or fire brigade support) its services are partially restricted. The basic safety hard core includes mechanical, emergency brakes.

Because permanent blocking or too frequently repeated blocking is inappropriate, the final implementation must employ also fault avoidance techniques. The other reason for the fault avoidance application stems from the fact that only approximated fail-stop implementation is possible. Moreover, the above described configurations create only skeleton carrying common fault-tolerant techniques see e.g. (Maxion et al., 1987). In short, while auxiliary hardware components maintain supply-voltage levels, input signals filtering, and timing, the software techniques, namely time redundancy or skip-frame strategy, deal with non-critical inputs and outputs.

4 MULTIPLE LIFT CONTROL SYSTEM

The second case study deals with the multiple lift control system based on a dedicated multiprocessor architecture (for more details see Sveda, 1997). An incremental measurement device for position evaluation, and position and speed control of a lift cabin in a lift shaft can demonstrate reusability. The applied application pattern, incremental measurement, means in this case the recognition and counting of rectangular impulses that are generated by an electromagnetic or photoelectric sensor/impulse generator, which is fixed on the bottom of the lift cabin and which passes equidistant position marks while moving along the shaft. That device communicates with its environment through interfaces with impulse generator and drive controller. So, the first input, I, provides the values 0 or 1 that are altered with frequency equivalent to the cabin speed. The second input, D, provides the values "up," "down," or "idle." The output, P, provides the actual absolute position of the cabin in the shaft.

4.1 Two-level Structure for Lift Control

The next employed application pattern is the two-level structure: the higher level behaves as an event-driven component, which behavior is roughly described by the state sequence

initialization → position_indication → fault_indication

and the lower level, which behaves as a set of real-time interconnected components. The specification of the lower level can be developed by refining the higher level state "position_indication" into three communicating lower level automata: two noise-tolerant impulse detectors and one checking reversible counter.

4.2 Incremental Measurement for Position and Speed Control

The first automaton models the noise-tolerant impulse detector in the same manner as in previous case, see the following timed-state sequence:

$$(0, q_1) \overset{inp=0}{\rightarrow} \ldots \overset{inp=0}{\rightarrow} (i, q_1) \overset{inp=1}{\rightarrow} (i+1, q_2) \overset{inp=0}{\rightarrow} \ldots \overset{inp=0}{\rightarrow} (j, q_2) \ldots$$
$$\ldots \overset{inp=1}{\rightarrow} (k, q_{b/2+1}) \overset{inp=1}{\rightarrow} \ldots$$
$$\ldots \overset{inp=1}{\rightarrow} (m, q_{b-1}) \overset{inp=0}{\rightarrow} (m+1, q_b) \overset{inp=1}{\rightarrow} \ldots \overset{inp=1}{\rightarrow} (n, q_b) \overset{inp=0/IMP}{\rightarrow} (n+1, q_1)$$

i, j, k, m, n are integers representing discrete time instances in increasing order.

The information about a detected impulse is sent to the counting automaton that can also access the indication of the cabin movement direction through the input D. For the sake of fault-detection requirements, the impulse generator and the impulse transfer path are doubled. Consequently, a second, identical noise-tolerant impulse detector appears necessary. The subsequent application pattern is the checking reversible counter, which starts with the value (h + l)/2 and increments or decrements the value according to the “impulse detected” outputs

Table 1: Application patterns hierarchy.

fault management based on fail-stop behavior approximations
two-level (event-driven/real-time) structure
incremental measurement
noise-tolerant impulse detector checking reversible counter /O watchdog

from the first or second recognition automaton. Overflow or underflow of the preset values of h or l indicates an error. This detection process sends a message about a detected impulse and the current direction to the counting automaton, which maintains the actual position in the shaft. To check the counter, an I/O watchdog application pattern employs again a reversible counter that can compare the impulses from the sensor/impulse generator and the changes of the total value.

4.3 Lift Fault Management

The approach used accomplishes a consequent application pattern, fault management based on fail-stop behavior approximations, both in the form of (a) hazardous state reachability control and (b) hazardous state maintenance. In safe states, the lift cabins are fixed at any floors. The system is allowed to reach any hazardous state when all relevant processors have successfully passed the start-up checks of inputs and monitored outputs and of appropriate communication status. The hazardous state maintenance includes operational checks and consistency checking for execution processors. To comply with safety-critical conception, all critical inputs and monitored outputs are doubled and compared. When the relevant signals differ, the respective lift is either forced (with the help of a substitute drive if the shaft controller is disconnected) to reach the nearest floor and to stay blocked.

The basic safety hard core includes mechanical, emergency brakes. Again, more detailed specification should reflect not only safety but also functionality with fault-tolerance support: also blocked lift is safe but useless. Hence, the above described configurations create only skeleton carrying common fault-tolerant techniques.

5 APPLICATION PATTERNS REUSE

The two case studies presented above demonstrate the possibility to reuse effectively substantial parts of the design dealing with petrol pumping station technology for a lift control technology project. While both cases belong to embedded control systems, their application domains and their technology principles differ: volume measurement and dosage control seems not too close to position measurement and control. Evidently, the similarity is observable by employment of application patterns hierarchy, see Table 1.

The reused upper-layer application patterns presented include the automata-based descriptions of incremental measurement, two-level (event-driven/real-time) structure, and fault management stemming from fail-stop behavior approximations. The reused lower-layer application patterns are exemplified by the automata-based descriptions of noise-tolerant impulse detector, checking reversible counter, and I/O watchdog.

Clearly, while all introduced application patterns correspond to design patterns in the above-explained interpretation, the upper-layer application patterns can be related also to frameworks. Moreover, the presented collection of application patterns creates a base for a pattern language supporting reuse-oriented design process for real-time embedded systems.

6 KNOWLEDGE-BASED SUPPORT

Industrial scale reusability requires a knowledge-based support, e.g. by case-based reasoning (see Kolodner, 1993), which differs from other rather traditional methods of Artificial Intelligence relying on case history. For a new problem, the case-based

reasoning strives for a similar old solution. This old solution is chosen according to the correspondence of a new problem to some old problem that was successfully solved by this approach. Hence, previous significant cases are gathered and saved in a case library. Case-based reasoning stems from remembering a similar situation that worked in past. For software reuse, case-based reasoning utilization has been studied from several viewpoints as discussed e.g. by Henninger (1998), and by Soundarajan and Fridella (1998).

6.1 Case-based Reasoning

The case-based reasoning method contains (1) elicitation, which means collecting those cases, and (2) implementation, which represents identification of important features for the case description consisting of values of those features. A case-based reasoning system can only be as good as its case library: only successful and sensibly selected old cases should be stored in the case library. The description of a case should comprise the corresponding problem, solution of the problem, and any other information describing the context for which the solution can be reused. A feature-oriented approach is usually used for the case description.

Case library serves as the knowledge base of a case-based reasoning system. The system acquires knowledge from old cases while learning can be achieved accumulating new cases. While solving a new case, the most similar old case is retrieved from the case library. The suggested solution of the new case is generated in conformity with this retrieved old case. Search for the similar old case from the case library represents important operation of case-based reasoning paradigm.

6.2 Backing Techniques

Case-based reasoning relies on the idea that situations are mostly repeating during the life cycle of an applied system. Further, after some period, the most frequent situations can be identified and documented in the case library. So, the case library can usually cover common situations. However, it is impossible to start with case-based reasoning from the very beginning with an empty case library.

When relying on the case-based reasoning exclusively, also the opposite problem can be encountered: after some period the case library can become huge and very semi-redundant. Majority of registered cases represents clusters of very similar situations. Despite careful evaluation of cases before saving them in the case library, it is difficult to avoid this problem.

In an effort to solve these two problems, the case-base reasoning can be combined with some other paradigm to compensate these insufficiencies. Some level of rule-based support can partially cover these gaps with the help of rule-oriented knowledge; see Sveda, Babka and Freeburn (1997).

Rule-based reasoning should augment the case-based reasoning in the following situations:

- No suitable old solution can be found for a current situation in the case library and engineer hesitates about his own solution. So, rule-based module is activated. For a very restricted class of tasks, the rule-based module is capable to suggest its own solution. Once generated by this part of the framework, such a solution is then evaluated and tested more carefully. However, if the evaluation is positive, this case is later saved in the case library covering one of the gaps of the case-based module.
- Situations are similar but rarely identical. To fit closer the real situation, adaptation of the retrieved case is needed. The process of adaptation can be controlled by the rule-based paradigm, using adaptation procedures and heuristics in the form of implication. Sensibly chosen meta-rules can substantially improve the effectiveness of the system.

The problem of adaptation is quite serious when a cluster of similar cases is replaced by one representative only - to avoid a high level of redundancy of the case library. The level of similarity can be low for marginal cases of the cluster. So, adaptation is more important here.

Three main categories of rules can be found in the rule-based module:

- Several *general heuristics* can contribute to the optimal solution search of a very wide class of tasks.
- However, the dominant part of the knowledge support is based on a *domain-specific rule*.
- For a higher efficiency, *metarules* are also attached to the module. This “knowledge about knowledge” can considerably contribute to a smooth reasoning process.

While involvement of an expert is relatively low for case-based reasoning module, the rules are mainly based on expert’s knowledge. However, some pieces of knowledge can also be obtained by data mining.

6.3 Similarity Measurement of State-based Specifications

Retrieval schemes proposed in the literature can be classed based upon the technique used to index cases during the search process (Atkinson, 1998): (a) classification-based schemes, which include keyword or feature-based controlled vocabularies; (b) structural schemes, which include signature or structural characteristics matching; and (c) behavioral schemes; which seek relevant cases by comparing input and output spaces of components.

The problem to be solved arises how to measure the similarity of state-based specifications for retrieval. Incidentally, similarity measurements for relational specifications have been resolved by Jilani, et al. (2001). The primary approach to the current application includes some equivalents of abstract data type signatures, belonging to structural schemes, and keywords, belonging to classification schemes. While the first alternative means for this purpose to quantify the similarity by the topological characteristics of associated finite automata state-transition graphs, such as number and placement of loops, the second one is based on a properly selected set of keywords with subsets identifying individual patterns. The current research task of our group focuses on experiments enabling to compare those alternatives.

7 CONCLUSIONS

The original contribution of this paper consists in proposal how to represent a system's formal specification as an application pattern structure of specification fragments. Next contribution deals with the approach how to measure similarity of formal specifications for retrieval in frame of case-based reasoning support. The above-presented case studies, which demonstrate the possibility to effectively reuse concrete application pattern structures, have been excerpted from two realized design cases.

The application patterns, originally introduced as "configurations" in the design project of petrol pumping station control technology based on multiple microcontrollers (Sveda, 1996), were effectively -- but without any dedicated development support -- reused for the project of lift control technology (Sveda, 1997). The notion of application pattern appeared for the first time in (Sveda, 2000) and developed in (Sveda, 2006). By the way, the first experience of the authors with case-based reasoning support to knowledge preserving development of an industrial application was published in (Sveda, Babka and Freeburn, 1997).

ACKNOWLEDGEMENTS

The research has been supported by the Czech Ministry of Education in the frame of Research Intention MSM 0021630528: Security-Oriented Research in Information Technology, and by the Grant Agency of the Czech Republic through the grants GACR 102/05/0723: A Framework for Formal Specifications and Prototyping of Information System's Network Applications and GACR 102/05/0467: Architectures of Embedded Systems Networks.

REFERENCES

Alexander, C. (1977) A Pattern Language: Towns / Buildings / Construction, Oxford University Press.

Alur, R. and T.A. Henzinger (1992) Logics and Models of Real Time: A Survey. In: (de Bakker, J.W., et al.) Real-Time: Theory in Practice. Springer-Verlag, LNCS 600, 74-106.

Arora, A. and S.S. Kulkarni (1998) Component Based Design of Multitolerant Systems. IEEE Transactions on Software Engineering, 24(1), 63-78.

Atkinson, S. (1998) Modeling Formal Integrated Component Retrieval. Proceedings of the Fifth International Conference on Software Reuse, IEEE Computer Society, Los Alamitos, California, 337-346.

Coad, P. and E.E. Yourdon (1990) Object-Oriented Analysis, Yourdon Press, New York.

Frakes, W.B. and K. Kang (2005) Software Reuse Research: Status and Future. IEEE Transactions on Software Engineering, 31(7), 529-536.

Gamma, E., R. Helm, R. Johnson and J. Vlissides (1995) Design Patterns -- Elements of Reusable Object-Oriented Software, Addison-Wesley.

Geppert, B. and F. Roessler (2001) The SDL Pattern Approach – A Reuse-driven SDL Design Methodology. Computer Networks, 35(6), Elsevier, 627-645.

Henninger, S. (1997) An Evolutionary Approach to Constructing Effective Software Reuse Repositories. Transactions on Software Engineering and Methodology, 6(2), 111-140.

Henninger, S. (1998) An Environment for Reusing Software Processes. Proceedings of the Fifth International Conference on Software Reuse, IEEE Computer Society, Los Alamitos, California, 103-112.

Holtzblatt, L.J., R.L. Piazza, H.B. Reubenstein, S.N. Roberts and D.R. Harris (1997) Design Recovery for

Distributed Systems. IEEE Transactions on Software Engineering, 23(7), 461-472.
Jacobson, L. (1992) Object-Oriented Software Engineering: A User Case-Driven Approach, ACM Press.
Jilani L.L., J. Deshamais and A. Mili (2001) Defining and Applying Measures of Distance Between Specifications. IEEE Transactions on Software Engineering, 27(8), 673-703.
Johnson, R.E. (1997) Frameworks = (Components + Patterns), Communications of the ACM, 40(10), 39-42.
Kolodner, J. (1993) Case-based Reasoning, Morgan Kaufmann, San Mateo, CA, USA.
Mili, R., Mili, A. and Mittermeir, R.T. (1997) Storing and Retrieving Software Components: A Refinement Based System. IEEE Transactions on Software Engineering, 23(7), 445-460.
Sen, A. (1997) The Role of Opportunity in the Software Reuse Process. IEEE Transactions on Software Engineering, 23(7), 418-436.
Shaw, M. and D. Garlan (1996) Software Architecture, Prentice Hall.
Soundarajan, N. and S. Fridella (1998) Inheritance: From Code Reuse to Reasoning Reuse. Proceedings of the Fifth International Conference on Software Reuse, IEEE Computer Society, Los Alamitos, California, 206-215.
Sutcliffe, A. and N. Maiden (1998) The Domain Theory for Requirements Engineering. IEEE Transactions on Software Engineering, 24(3), 174-196.
Sveda, M. (1996) Embedded System Design: A Case Study. IEEE Proc. of International Conference and Workshop ECBS'96, IEEE Computer Society, Los Alamitos, California, 260-267.
Sveda, M., O. Babka and J. Freeburn (1997) Knowledge Preserving Development: A Case Study. IEEE Proc. of International Conference and Workshop ECBS'97, Monterey, California, IEEE Computer Society, Los Alamitos, California, 347-352.
Sveda, M. (1997) An Approach to Safety-Critical Systems Design. In: (Pichler, F., Moreno-Diaz, R.) Computer Aided Systems Theory, Springer-Verlag, LNCS 1333, 34-49.
Sveda, M. (2000) Patterns for Embedded Systems Design. In: (Pichler, F., Moreno-Diaz, R., Kopacek, P.) Computer Aided Systems Theory--EUROCAST'99, Springer-Verlag, LNCS 1798, 80-89.
Sveda, M. and R. Vrba (2006) Fault Maintenance in Embedded Systems Applications. Proceedings of the Engineering of Computer-Based Systems. Proceedings of the Third International Conference on Informatics in Control, Automation and Robotics (ICINCO 2006), INSTICC, Setúbal, Portugal, 183-186.
Turner, K.J. (1997) Relating Architecture and Specification. Computer Networks and ISDN Systems, 29(4), 437-456.
van Lamsweerde, A. and L. Willemet (1998) Inferring Declarative Requirements Specifications from Operational Scenarios. IEEE Transactions on Software Engineering, 24(12), 1089-1114.
Xinyao, Y., W. Ji, Z. Chaochen and P.K. Pandya (1994) Formal Design of Hybrid Systems. In: (Langmaack, H., W.P. de Roever and J. Vytopil) Formal Techniques in Real-Time and Fault-Tolerant Systems, Springer-Verlag, LNCS 863, 738-755.
Zaremski, A.M. and J.M. Wing (1997) Specification Matching of Software Components. ACM Trans. on Software Engineering and Methodology, 6(4), 333-369.

A PARAMETERIZED GENETIC ALGORITHM IP CORE DESIGN AND IMPLEMENTATION

K. M. Deliparaschos, G. C. Doyamis and S. G. Tzafestas
School of Electrical and Computer Engineering, National Technical University of Athens
Iroon Polytechneiou 9,15780, Zographou Campus, Athens, Greece
kdelip@mail.ntua.gr, gdoyamis@ieee.org, tzafesta@softlab.ntua.gr

Keywords: Genetic Algorithm (GA), Travelling Salesman Problem (TSP), Field Programmable Gate Array (FPGA) chip, Very High–speed Integrated Circuits Description Language (VHDL), Intellectual Property (IP) core.

Abstract: Genetic Algorithm (GA) is a directed random search technique working on a population of solutions and based on natural selection. However, its convergence to the optimum may be very slow for complex optimization problems, especially when the GA is software implemented, making it difficult to be used in real time applications. In this paper a parameterized GA Intellectual Property (IP) core is designed and implemented on hardware, achieving impressive time–speedups when compared to its software version. The parameterization stands for the number of population individuals and their bit resolution, the bit resolution of each individual's fitness, the number of elite genes in each generation, the crossover and mutation methods, the maximum number of generations, the mutation probability and its bit resolution. The proposed architecture is implemented in a Field Programmable Gate Array Chip (FPGA) with the use of a Very-High-Speed Integrated Circuits Hardware Description Language (VHDL) and advanced synthesis and place and route tools. The GA discussed in this work achieves a frequency rate of 92 MHz and is evaluated using the Traveling Salesman Problem (TSP) as well as several benchmarking functions.

1 INTRODUCTION

Genetic Algorithms (GAs), initially developed by Holland (Holland 1975), are based on the notion of *population individuals* (*genes/chromosomes*), to which genetic operations as mutation, crossover and elitism are applied. Genetic algorithms obey Darwin's natural selection law i.e., the survival of the fittest. GAs have been successfully applied to several hard optimization problems, due to their endogenous flexibility and freedom in finding the optimal solution of the problem (Mitchell 1996, Goldberg 1989).

However, the most serious drawbacks of software-implemented GAs are both the vast time and system resources consumption. Keeping that in mind, a multitude of hardware-implemented GAs have been evolved mainly during the last decade, exploiting the rapid evolution in the field of the Field Programmable Gate Arrays (FPGAs) technology and achieving impressive time-speedups.

This paper deploys the design and hardware implementation of a parameterized GA Intellectual Property (IP) core (*Semiconductor intellectual property core* n.d.) on an FPGA chip. The genetic operators applied to the genes of the population are *crossover*, *mutation* and *elitism*, whose employed method is parametrically selected. The designed selection algorithm is the "*Roulette Wheel Selection Algorithm*". The FPGA chip used in this work is a Xilinx XC3S1500–4FG676C Spartan-3 FPGA (Xilinx 2003). A software implementation of the designed GA using the Matlab Platform has also been developed to produce input and output test vectors for the performance evaluation of the hardware implemented GA using several benchmark functions, described below. Finally, after adapting the proposed hardware implemented GA to the Traveling Salesman Problem (TSP), a successful solution to it has been found.

2 GA HARDWARE ARCHITECTURE

This section describes and explains analytically the various hierarchical modules of the presented GA architecture.

2.1 System Overview

A high level architectural structure of the system is shown in Figure 1.

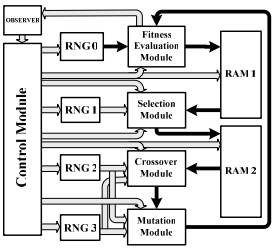


Figure 1: High level architectural structure.

2.2 GA Characteristics

Table 1: GA Characteristics.

Parameter name	Description
genom_lngt	Chromosome length in bits
score_sz	Fitness value bit resolution
pop_sz	Population size
scaling_factor_res	Bit resolution of the random number used in RWS algorithm
elite	Number of elite offsprings
mr	Mutation rate
mut_res	Bit resolution of the random number used in mutation
fit_limit	Fitness limit
max_gen	Maximum generations number
inv_type	Type of the inversion of the fitness value (used only in TSP) 1: division 2: subtraction

2.3 GA Architecture

As shown in Figure 1, the architecture is broken into separate blocks each one of which performs a particular task, coordinated by the control block. Moreover, they send back signals to the control module notifying their state i.e., ready out signals.

2.3.1 Control Module

In order to assure, control and synchronize the order of execution of the several hardware implemented modules of the proposed GA architecture, a control module has been implemented for that reason. This block produces and feeds all other modules with the needed control signals using a nine-state Mealy state machine (Zainalabedin 1998). The task performed by each of the nine states, is described in Table 2.

Table 2: Diverse states of the implemented state machine.

State	Description
clear_ram	Clear RAM 1
fill_ram	Fill RAM 1 with a random gene to create the initial population
fit_eval	Fitness evaluation of the input gene and generation of the elite offsprings' indexes
sel	Selects one parent among the genes of the current population
cross	Apply crossover operation
mut	Apply mutation operation
done	Check of the termination criteria
read_write_ram_1	Read/write RAM 1
read_write_ram_2	Read/write RAM 2

2.3.2 Fitness Evaluation Module

This section describes the fitness evaluation module, which functions every time a new population of individuals is formed. This block performs two separate tasks; on the one hand it calculates the fitness of each individual according to the given fitness function and, on the other, it performs elitism on the current population producing the elite genes for the next generation. Having that in mind, the structural architecture of this module consists of two sub-modules as shown in Figure 2.

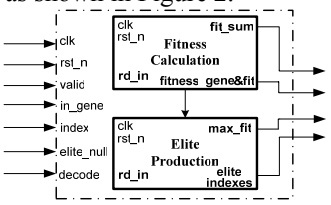


Figure 2: High level architectural structure.

The one is used for fitness calculation and the other for performing elitism. The above mentioned modules are externally connected to the remaining GA architecture, so as to allow any further fitness functions without affecting the rest of the design. Furthermore, the module outputs both the sum of fitnesses and the maximum fitness of the current population as well as the RAM indexes of the elite genes. The number of elite genes is parametrically set. In order for an individual to become an elite gene of the next generation, i.e., to survive in the next generation, it has to be the fittest among the rest individuals.

2.3.3 Selection Module

This section describes the selection module, which operates after the fitness evaluation of the individuals

of the current population has ended and is controlled by the control module. The selection block, connected to both RAMs, implements the Roulette Wheel Selection Algorithm (RWS) (Koza *et al.* 1999, Mitchell 1996).

The selection block will keep on selecting parents till the number of them suffices to produce a new population with an equal number of individuals like the one in the current population.

2.3.4 Crossover Module

This section describes the crossover module, which runs after the selection module has completed its task and applies the crossover operation to the selected parents. The crossover method to be implemented is parametrically employed. There are diverse crossover strategies reported in literature (Koza *et al.* 1999, Tzafestas 1999). The present implementation includes three different crossover methods, i.e., single point, two point and uniform crossover.

The crossover module needs a couple of random numbers (random crossover points, random mask), according to the method employed, in order to apply the desired crossover operation. As a result two random number generators are used for that reason. The former produces the crossover points and the other the mask needed for the application of uniform crossover to the parents. The crossover block outputs one offspring in each execution, produced by two of the selected parents.

2.3.5 Mutation Module

This section describes the mutation module, which functions after the crossover module has completed its task and applies the parametrically employed mutation method to the crossovered offsprings, i.e., the offsprings produced by crossover module. Various mutation strategies are reported in literature (Koza 1999, Tzafestas 1999). The proposed design includes three different mutation methods, i.e., single point, masked and uniform mutation.

The mutation module requires a couple of random numbers (random mutation points, random mask, random numbers $p_{r,i}$), according to the method employed so as to apply the desired mutation operation. For this reason two random number generators are needed. The former produces the mutation points and the other a random binary mask. The latter RNG also generates the necessary random number p_r, in order to decide if mutation operation will be applied, i.e., only if its value is less or equal to the parametrically set mutation probability p_m, will mutation be applied to the offsprings. If the mutation method employed is uniform mutation, it also generates the essential random numbers for uniform mutation. The mutation block outputs one offspring in each execution produced by processing one offspring, result of crossover, each time.

2.3.6 Observer Module

This section describes the observer module, which executes each time a new population has been formed. This block determines the continuation of the algorithm, checking if the parametrically set stopping criteria, i.e., maximum generations, fitness value limit, have been met.

2.3.7 Random Number Generators

This section describes the random number generator modules, which feed most of the described modules above with random numbers. This block implements a Linear Feedback Shift Register (LFSR) generator, whose sequence length is parametrically set. Four random number generators (RNG) are used to produce both the initial random generation and the necessary random numbers. The maximum length of the random generated sequence is 128 bits, while the specific characteristics of the four RNGs used in our design are shown in Table 3.

Table 3: Characteristics of the used RNGs.

RNG	LFSR length
RNG 0	genom_lngt
RNG 1	scaling_factor_res
RNG 2	genom_lngt + mut_res
RNG 3	2•log2(genom_lngt)

2.3.8 Random Access Memory (RAM)

This section describes the Random Access Memory (RAM) modules, RAM 1 and RAM 2, which store the current population and the selected parents respectively. Both the address and data widths are parametrically set, as shown in Table 4.

Table 4: Parameters of RAM 1 and RAM 2.

RAM	Address width	Data width
RAM 1 (population RAM)	genom_lngt	genom_lngt + score_sz
RAM 2 (parents RAM)	2•(pop_sz-elite)	genom_lngt

3 DESIGN FLOW FOR THE GA

This section presents the design flow, illustrated in Figure 3, in a top-down manner (Deliparaschos *et al.* 2006), followed in our design. In a top-down design, one first concentrates on specifying and then on

designing the circuit functionality (Sjoholm *et al.* 1997). The starting point of the design process is the system level modelling of the proposed GA, so as to evaluate the proposed model and to extract valuable test vector values to be used later for RTL and timing simulation. Special attention has been paid on the coding of the different blocks, since we aim at writing a fully parameterized GA code. An RTL simulation has been performed to ensure the correct functionality of the circuit. Next, logic synthesis has been done, where the tool first creates a generic (technology-independent) schematic on the basis of the VHDL code and then optimizes the circuit to the FPGA specific library chosen (Spartan-3 1500-4FG676). At this point, area and timing constraints and specific design requirements must be defined as they play an important role for the synthesis result.

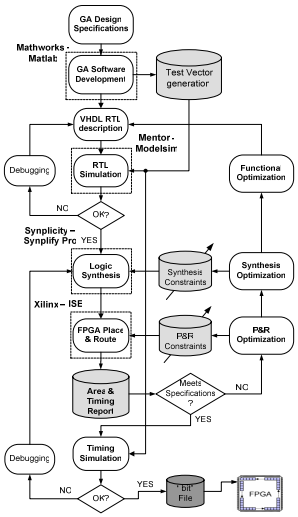


Figure 3: Design Flow.

Following, the Xilinx ISE place and route (PAR) tool accepts the input netlist file (.edf), generated with Synplify Pro synthesis tool, translates and maps our design on the FPGA device. Finally, it places and routes the FPGA producing output for the bitstream generator (BitGen). The latter program generates a bitstream (.bit) for Xilinx device configuration. Before programming the FPGA file, a timing simulation is performed to ensure that the circuit meets the timing requirements set and works correctly.

4 IMPLEMENTATION RESULTS

This section describes the implementations results of the designed GA. After the synthesis of the design, ISE translates, maps, and places and routes our design to the FPGA device. The FPGA utilization for both the GA and the GA adapted to the TSP produced by ISE are shown in Table 5.

The hardware implementation of the proposed GA achieves an internal clock frequency rate of 92 MHz (10,8 ns) while the adapted GA to the TSP achieves an internal clock frequency rate of 91 MHz (11 ns). Moreover, 2.450 ns (2,4 μs) and 14.391 ns (14,3 μs) are required to form a new generation of 8 individuals in the former and latter version of the GA respectively. Finally, the VHDL codes for the GA models presented here are fully parameterized, allowing us to generate and test the GA models with different specification scenarios.

Table 5: FPGA Utilization for the implemented GAs.

Logic Utilization	GA	GA (adapted to TSP)
Slice flip flops	681 (2%)	1045 (3%)
4 – Input LUTs	1086 (4%)	1630 (6%)
Logic Distribution		
Occupied slices	892 (6%)	1305 (9%)
4 – Input LUTs	1116 (6%)	1686 (6%)
Used as logic	1086	1630
Used as route-thru	6	4
Used as 16x1 ram	24	52
Bonded IOBs	59 (12%)	53 (10%)
MULT 18x18s	1 (3%)	3 (9%)
GCLKs	1 (12%)	1 (12%)

5 EVALUATION RESULTS

The evaluation of the system performance has been made both by solving the TSP problem and by optimizing several benchmark functions (Digalakis *et al.* 2000, Zhang and Zhang 2000), which are noted in section 5.2.2. In order to evaluate the performance of the implemented GA using the TSP, we firstly have to adapt the hardware to the TSP (section 5.1) and secondly to write a software version of the hardware implemented GA, which will also be adapted to the TS problem. Both the software version of the GA and the one adapted to the TSP have been developed on Matlab Platform.

5.1 GA Hardware Adaptation to the TSP

According to the definition of the TSP (Pham and Karaboga 2000), each city should be visited only once. So every gene of the population, which contains the towns to be visited in sequence, must contain each town only once. Since our genes are unique, we cannot possibly use the above mentioned crossover

and mutation techniques, but new crossover and mutation methods must be developed (explained below) (Martel 2006).

The crossover operator uses a pool of indices of the towns. In order to keep the uniqueness of the visited cities used indices will be removed from the pool. The offspring produced after the application of crossover to the parents, is formed via the following procedure: In the beginning a random crossover point is generated and then the town indices of the first parent are added to the offspring starting from the crossing site, to the head of the gene. The afore-mentioned indices are removed from the pool. Afterwards the right side of the gene is filled by checking whether the town indices in the right side of the second parent are contained in the pool. If a town index is still free, i.e. exists both in the pool and the second parent, we place it in the offspring and remove it from the pool; otherwise we skip it and leave its place in the offspring empty till we reach the tail of the gene. Finally empty places of the offspring are randomly filled with the town indices remaining in the pool. The developed mutation operator utilizes two random generated mutation points and simply swaps the town indices stored at these points of the gene. The described methods for 8 cities are depicted in Figure 4.

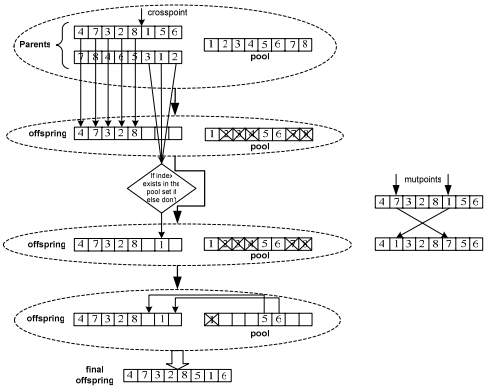


Figure 4: Crossover and mutation operations for TSP.

The fitness evaluation function implemented here, is the computation of the sum over the number of towns (N) of the square of the Euclidean distances between two adjacent towns according to the computed tour, as shown in equation (1). We compute the square of the Euclidean distance in order to avoid the hardware implementation of the square root, since it is not necessary as we do nothing but merely compare the fitnesses among the chromosomes.

$$fit_j = \sum_{i=1}^{N-1}\left(\sqrt{(x_i - x_{i+1})^2 + (y_i - y_{i+1})^2}\right)^2 + \left(\sqrt{(x_N - x_1)^2 + (y_N - y_1)^2}\right)^2 \quad j \in [1, pop\_sz] \tag{1}$$

Since we seek the optimal minimum path connecting the given towns and the GA is designed to result in the gene with the maximum fitness instead, i.e., with the maximum sum of Euclidean distances, we have either to invert the computed fitness or to subtract the fitness from its highest value, according to the resolution of bits adopted for the binary coding of it, i.e. 2^{score_sz}-1 (See Table 1), in order to get higher values for smaller path lengths. The method for the inversion of the fitness to be implemented is selected through a generic in the VHDL code (*inv_type*).

5.2 Results

The performance evaluation of the proposed GA using the TSP and various benchmark functions follow.

5.2.1 TSP

The performance evaluation of the proposed GA using the TSP has been performed by comparing the time needed for the software version developed and the one needed for the hardware implementation to find the optimal solution. The results for eight cities, 60 generations and 32 individuals are summarized in Table 6, where an impressive speedup ratio of 11.035 can be observed. Figure 5 depicts the map of the 8 cities used, which are existing cities of the greek territory. The algorithm was also tested using the benchmark burma14 derived from the TSPLIB (*Gerhard* n.d.), and the result is depicted in Figure 6.

Table 6: Software vs. Hardware GA.

GA version	Time (msec)
Hardware (11 nsec)	1,702
Software (Pentium 4 3,2 GHz 1Gb RAM)	18.783

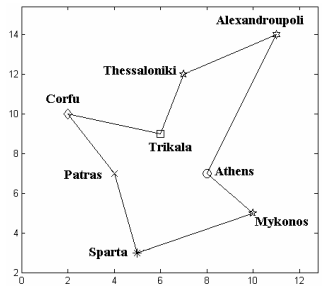


Figure 5: Map of the cities used in the TSP.

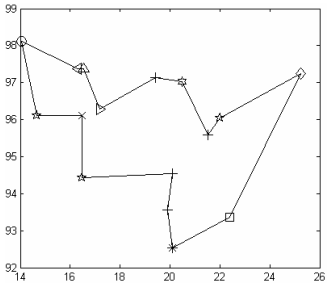


Figure 6: TSP solution of burma14 benchmark.

5.2.2 Benchmark Functions

Several benchmark functions are known in the literature (Digalakis and Margaritis 2000, Zhang *et al.* 2000) for evaluating the performance of a GA, i.e. its ability to reach the optimum of an objective function. In our case we have tested the proposed GA using the following functions, which are noted in Table 7 and depicted in figures 7–9.

Table 7: Benchmarking functions.

Name	Function type
F1: Zhang Zhang	$\left(\mathbf{1}-\mathbf{2}\sin^{\mathbf{20}}\left(\mathbf{3}\pi x\right)+\sin^{\mathbf{20}}\left(\mathbf{20}\pi x\right)\right)^{\mathbf{20}}$ $x\in\left(\mathbf{0},\mathbf{1}\right)$
F2: Rastrigin	$\mathbf{100}-\sum_{i=\mathbf{1}}^{\mathbf{2}}\left(x_i^{\mathbf{2}}-\mathbf{10}\cos\left(\mathbf{2}\pi x_i\right)\right)$ $-\mathbf{5},\mathbf{12}\le x_i\le\mathbf{5},\mathbf{12}$
F3: Easom	$\left(\prod_{i=\mathbf{1}}^{\mathbf{2}}\cos\left(x_i\right)\right)\times\left(e^{-\sum_{i=\mathbf{1}}^{\mathbf{2}}\left(x_i-\pi\right)^{\mathbf{2}}}\right)$ $-A\le x_i\le A$ $A\in\left(\mathbf{10},\mathbf{100}\right)$

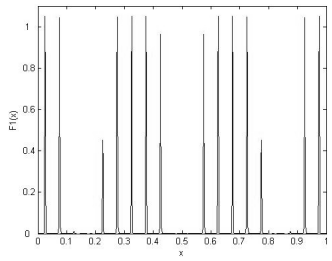


Figure 7: Zhang Zhang function (F1).

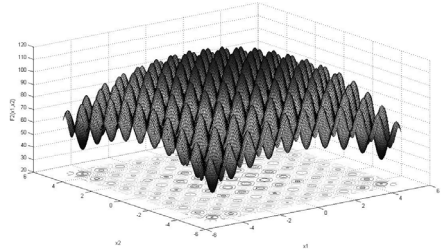


Figure 8: Rastrigin function (F2).

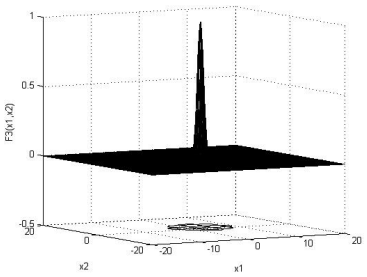


Figure 9: Easom function (F3).

In the following figures the results of a number of experiments using the above mentioned functions are presented. Figure 10 shows the effect of the chromosome length (*genom_lngt*) on the optimal solution found, while Figure 11 depicts the effect of the population size (*pop_sz*) on the generations needed by the algorithm to converge. Finally the influence of the population size on the calculation time is shown in Figure 12. We also have to note that the precision of the optimum value found by the proposed GA depends on the chromosome length adopted in each experiment. A length of 16 and 32 bits for evaluating one and two-variable benchmark functions, respectively, is observed to give high accuracy on the result in relatively low calculation times.

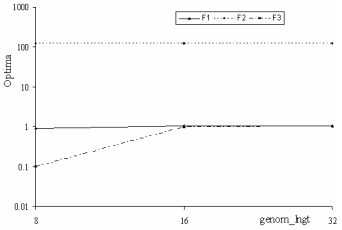


Figure 10: Estimated optima vs. chromosome length.

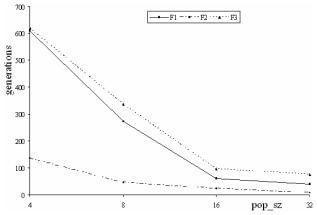


Figure 11: Estimated generations vs. population size.

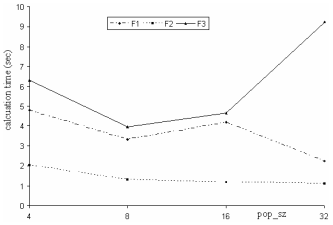


Figure 12: Estimated calculation time vs. population size.

6 CONCLUSION

In this work we presented a fully parameterized Genetic Algorithm IP in terms of the number of population individuals (*pop_sz*) and their resolution in bits (*genom_lngt*), resolution in bits of the fitness (*score_sz*), number of elite genes in each generation (*elite*), method used for crossover (cross_method) and mutation (mut_method), number of maximum generations (*max_gen*), mutation probability (*mr*) and its resolution in bits (*mut_res*), as well as the resolution in bits of the scaling factor used by the RWS algorithm. This parameterization allows the adaptation of the GA to any problem specifications without any further change to the developed VHDL code. Furthermore, the proposed hardware implemented GA operates at a clock rate of 92 MHz (10,8 ns) and achieves a noteworthy speedup when compared to its software version. Additionally, the hardware area required for the implementation and the requirements of RAM are kept small according to the PAR report (see Table 5). Compared to other GAs hardware implementations (Zhu *et al.* 2006, Aporntewan and Chongstitvatana 2001, Lei *et al.* 2002, Tang and Yip 2002), our design operates at a clock frequency up to five times faster and implements more than one crossover and mutation methods, which can be changed during its execution. Moreover, our design utilizes more parameters and is evaluated not only by using benchmarking functions but also by solving the NP-complete Travelling Salesman Problem.

REFERENCES

Aporntewan, C., Chongstitvatana, P., 2001. A hardware implementation of the Compact Genetic Algorithm. *Proceedings of the 2001 Congress on Evolutionary Computation,* 1, pp. 624–629.

Deliparaschos, K.M., Nenedakis, F.I., Tzafestas, S.G., 2006. Design and implementation of a fast digital fuzzy logic controller using FPGA technology. *Journal of Intelligent and Robotics Systems,* 45, pp. 77–96.

Digalakis, J.G., Margaritis, K.G., 2000. An experimental study of benchmarking functions for genetic algorithms. *in: 2000 IEEE Int. Conference on Systems, Man, and Cybernetics,* 5, pp. 3810–3815.

Gerhard, R., *TSP libraries.* Department of Computer Sciences, University of Heidelberg. Available from: http://www.iwr.uniheidelberg.de/groups/comopt/software/TSPLIB95/

Goldberg, D.E., 1989. *Genetic Algorithms in Search Optimization and Machine Learning.* Reading, MA: Addison-Wesley.

Holland, J.H., 1975. *Adaptation in Natural and Artificial systems: An Introductory Analysis with Application to Biology, Control, and Artificial Intelligence.* Ann Arbor, MI: University of Michigan Press.

Koza, J.R., 1992. *Genetic Programming: On the programming of computers by means of natural selection.* Cambridge, MA: MIT Press.

Koza, J.R., Bennett III, F.H., Andre, D., Keane, M.A., 1999. Genetic *Programming III: Darwinian Invention and Problem Solving.* San Francisco, CA: Morgan Kaufmann Publishers.

Lei, T., Zhu M.-C., Wang J.-X., 2002. The hardware implementation of a genetic algorithm model with FPGA. *in: 2002 IEEE International Conference on Field–Programmable Technology (FPT),* pp. 374–377.

Mitchell, M., 1996. *An Introduction to Genetic Algorithms.* Cambridge, MA: MIT Press.

Martel, E., *Solving Travelling Salesman Problems using Genetic Algorithms.* Available from: http://ai-depot.com/Articles/51/TSP.html

Pham, D.T., Karaboga, D., 2000. *Intelligent Optimization Techniques: Genetic Algorithms, Tabu Search, Simulated Annealing and Neural Networks.* London, UK: Springer.

Sjoholm, S., Lindh, L., 1997. *VHDL for Designers.* London, UK: Prentice Hall.

Tang, W., Yip, L., 2004. Hardware implementation of genetic algorithms using FPGA. *in: MWSCAS '04, The 2004 47th Midwest Symposium on Circuits and Systems,* 1, pp. 549–552.

Tzafestas, S.G., 1999. *Soft Computing in Systems and Control Technology.* 18, London, UK: World Scientific.

Wikipedia, *Semiconductor intellectual property core.* Available from: http://en.wikipedia.org/wiki/Semiconductor_intellectual_property_core

Xilinx, 2003. *Spartan-3 FPGA Family: Complete Data Sheet - DS099.* Available from: http://www.xilinx.com/bvdocs/publications/ds099.pdf

Zainalabedin, N., 1998. *VHDL: Analysis and Modeling of Digital Systems.* NY: Mc Graw-Hill International.

Zhu, Z., Mulvaney, D., Chouliaras, V., 2006. A novel genetic algorithm designed for hardware implementation. *Int. Journal of Computational Intelligence,* 3, number 4.

Zhang, L., Zhang, B., 2000. Research on the mechanism of genetic algorithms. *Journal of Software,* 11(7), pp.945–952.

MORE EXPRESSIVE PLANNING GRAPH EXTENSION

Joseph Zalaket and Guy Camilleri
Département informatique, Université Saint-Esprit de Kaslik,B.P. 446 Jounieh, Liban
IRIT CSC, Université Paul Sabatier,118 route de Narbonne, 31062 Toulouse, France
josephzalaket@usek.edu.lb, camiller@irit.fr

Keywords: AI Planning, planning graph, numerical planning, functional planning.

Abstract: The most of the real world problems have characteristics that can not be handled like pure propositional expressions. Many classical planners have been extended to deal with more expressive problems which require features like temporal actions, numeric functions, conditional effects, ... etc. Extensions have been made to planners like (D. and D., 1999), MetricFF02, (Do and Kambhampati, 2001), which use the planning graph as their base structure. However, each of these extensions deals with the one or the other of the features required for more expressive planning problems without being able to integrate all or at least an important combination of these features. In this paper, we propose an extension to the planning graph structure to deal with more expressive planning problems that can contain real word data characteristics, such as: symbolic, temporal and numeric data.

1 INTRODUCTION

The domain independent-planning still a complex problem. The most of the domain-independent planners are able to solve symbolic planning problems but, real-world problems necessity dealing with more complex types of data such as temporal and numeric data. This kind of data requires features like numeric function and temporal actions to be added and supported by the planning process. Some planners have been extended to deal with some real world knowledge, and thus, to support one or the other of these features. Planners like SAPA (Do and Kambhampati, 2001) and Metric-FF (Hoffmann, 2002) have extended their heuristic search to deal with numeric knowledge. However, to combine multiple features which can be necessary to solve some more real-world problems in a domain-independent planner still a complex task. In the other hand, many planners are using the planning graph structure as their base for exploring the state space due to its compact structure. After its introduction in Graphlan (Blum and Furst, 1995) as its basic expansion graph, the planning graph became the subject of multiple extensions to solve directly some more expressive problems like temporal Graphplan (TGP) (Smith and Weld, 1999)or to serve as a base to calculate heuristic function for such problems like in like SAPA (Do and Kambhampati, 2001) and Metric-FF (Hoffmann, 2002). In this paper we propose a general extension to the planning graph, which allow it to deal with functions application. The functions that we propose can be incorporated into the action effects and/or the action preconditions. This extension allows the planning graph to support all types of features that are necessary to deal with more expressive planning problems, such as problems that contain numeric and temporal knowledge. In our extension we introduce the application of functions during the planning graph expansion, in a way that the results of these functions can be incorporated within the nodes of this graph and thus they can be treated as any other type of nodes. Using functions application will allow us to deal with any combination of features required for solving some real world problems as the functions that we propose can be written in any programming language without restrictions to their implementation. Therefore, the extended planning graph can be used directly as an expansion phase graph as in Graphlan-like planners (Blum and Furst, 1995), (Smith and Weld, 1999) to

solve planning problems or it can also be used as the base graph to calculate heuristics as in FF-like planners (Hoffman, 2001), (Hoffmann, 2002).

We first present the motivation for our work in section 2. We then present an overview of the planning graph structure in section 3. We present the extension to the definition of the planning problem in section 4. We explain the planning graph extension to handle numerical variables and functions application in section 5. We finally conclude this paper in section 6.

2 MOTIVATION

Many extensions have tried to adapt the planning graph to solve more expressive planning problems than those that can be expressed in pure propositional STRIPS language. Real-world problems have complex characteristics. Most of these characteristics can be modeled with the PDDL3.0 (A. and D., 2005) language which is the mostly used by recent planners. Therefore, PDDL3.0 gives the possibility of expressing optimization criterion such that maximization or minimization of resource consumption, without being able to represent all king of resource handling. The extension that we propose to the planning graph structure aims to allow this graph to supprot the execution of functions. Functional Strips (Geffner, 1999) and FSTRIPS (Schmid et al., 2000) have studied the integration of functions to solve planning problems. Functional Strips and FSTRIPS use functions instead of relations, which allow reducing the generated literals in a planning problem. In our work we introduce the function application which allow us to represent complex numerical and conditional effects, as well as complex preconditions and goals. For example, to plan the motion of a robot that is expected to travel from an initial position to a target position in the presence of physical obstacles in its environment. To avoid collisions between the robot and the obstacles we have to use trigonometric functions that allow the robot to follow a circular path in the environment (BAK M. and O., 2000). Trigonometric expressions are simple to be solved by classical programming languages, but they can not be expressed in classical AI planning languages like PDDL3.0. For this reason our extension to th planning graph is based in the integration of functions written in any programming language. Inspiring from the representation of functions in object-oriented databases. We add to the problem definition written in PDDL3.0 the declaration of external functions by specifying their execution path. In addition to their capabilities of solving numerical problems like the circular movement of a robot, external function are able to handle all kind of conditional and probabilistic effects. To avoid all kind of side effects we restrict all the formal parameters of a function to be constant, which mean a function can only return a value without updating any state variable. This returned value is affected to one state variable at a time.
The use of non-bijective functions requires that their application in the search space could be done only in forward. As the planning graph is constructed in progression, thus it can support the application of this kind of functions.

3 PLANNING GRAPH

Because The planning graph introduced in Graphplan (Blum and Furst, 1995) is the basis of our extension, we briefly summarize the planning graph structure without detailing how it is used by the Graphplan algorithm.

A planning graph consists of a sequence of levels, where level 0 corresponds to the initial state. Each level contains a set of literals and a set of actions. The actions of a level are those that have their preconditions satisfied and mutually consistent by the literals of the same level. The literals of a next level i+1 are the effects of the applicable actions of previous level i. The literals that satisfy the preconditions of action instances at the same level are connected with these actions via direct edges. Also, the action instances are connected via direct edges with their literals effects at the next level.

The planning graph is expanded up to a level in which all the literals of the goal are present and consistent. Then, a solution extraction phase can start in a way that for each sub-goal at the last level n, it tries to find an action instance at level n-1 that has this sub-goal as an add effect. The preconditions of the selected action instances become the new set of sub-goals at level n-1 and so on until reaching the initial state (level 0).

4 PLANNING PROBLEM EXTENSION

Each distinct instance of the world is called a state, denoted by s. The set of all possible states is called a state space S. S is formed by two disjoint sets: a set of logical propositions P and the set of numeric variables N, such that $S = \{(p,n)/p \in P, n : N \rightarrow \mathbb{R}\}$.

We denote by $P(s)$ the subset of logical propositions of a state s and by $N(s)$ the subset of its numeric variables.
A state transition functions t transforms s into $s' = t(s,a)$ by 3 ways:

- Adds logical propositions to $P(s)$
- Removes logical propositions from $P(s)$ (when propositions become false)
- Assigns new values to existing numeric variables in $N(s)$

An action a is defined as a tuple $\langle arg, pre, eff \rangle$, where:

- arg is the list of arguments made by variables which represent constant symbols and/or numeric variables in N.
- $pre = pre_P \cup pre_N$ is the list of preconditions.

  pre_P defined over P are propositional preconditions

  pre_N are numeric preconditions, such that: $\forall c \in pre_N$, $c = (n \theta g)$, where $n \in N$, $\theta \in \{<, \leq, =, >, \geq\}$ and g is an external function or an expression.
  Definition-1: *An expression is an arithmetic expression over N and the rational numbers, using the operators +, -, * and /.*
  Definition-2: *An external function is written in a high level programming language on the form of type $f(n_1, n_2, , n_m)$ where arguments $n_1, n_2, , n_m \in N$. The function and all its arguments (if exist) are declared as constants in a way that the function calculate and return a value without affecting any numeric state variable.*
- $eff = eff_P \cup eff_N$ is the list of effects.

  eff_P defined over P are positive and negative propositional effects that add or remove litterals.

  eff_N are numeric effects, such that: $\forall e \in eff_N$, $e = (n := g)$, where $n \in N$ and g is an external function or an expression.

A planning problem is defined as:

1. A nonempty state space S, which is a finite or countably infinite set of states.
2. For each state $s \in S$, a finite set of applicable actions $A(s)$.
3. A state transition function f that produces a state $s' = f(s,a) \in S$ for every $s \in S$ and $a \in A(s)$.
4. An initial state $s_I \in S$.
5. A set of goal conditions G satisfied in $s_G \in S$.

   The set of goal conditions $G = G_P \cup G_N$, where: G_P defined over P are propositional goals and G_N are numeric goals that should be satisfied in the goal state s_G, such that: $\forall l \in G_N$, $l = (n \theta g)$, where $n \in N$, $\theta \in \{<, \leq, =, >, \geq\}$ and g is an external function or an expression.

5 PLANNING GRAPH EXTENSION

In this section we present an extension to the planning graph structure to support numeric variables handling and external functions executions. We start the creation of the planning graph by instantiating the propositional variables of all the actions of the planning problem. This instantiation is done once before the graph expansion by substituting the propositional variables by all the combinations of the world objects. Therefore, none of numeric variables are substituted before the planning graph expansion process.// We initialize the fact level 0 to the initial state of the planning problem then we loop while the goal is not satisfied at the current fact level which is initially the level 0. For each one of the previously instantiated actions, we test first if all the propositional preconditions pre_P exist in the current level. If the first test is successful then we proceed to test the numerical conditions pre_N otherwise, we skip this second test. If the numerical preconditions pre_N of the action are satisfied in the current level we add this action to the applicable actions list of the current action level. For

rotate(Position p1, Position p2, robot r)
pre: $p1 \neq p2$, $at(r, p1)$, fcos(cont short θ) $\prec 1$
eff: $at(r, p2)$, $\neg at(r, p1)$, θ :=frotation(const short θ)

Figure 1: An action with external functions.

example, in the action rotate (fig.1) the first condition $p1 \neq p2$ is tested at the propositional instantiation of the actions as the condition concerns only world objects which are not covered by a relation (not a literal) or an external function. This kind of tests avoids instantiating non useful actions from the beginning. The second condition $at(r, p1)$ is a literal so it belongs to the propositional preconditions pre_P and it is tested first for choosing the applicable actions at the current level. Suppose that $at(r, p1)$ exists in the current fact level thus, the algorithm will proceed by testing the external function "fcos()" for the value of the formal parameter θ passed from the current fact level. If the value returned by "fcos()" is less than 1, then this ac-

tion will be added to the applicable actions of the current action level.

At each action level, the numeric variables of an action are instantiated just before the action application at this current level. This instantiation is done by replacing the variables by the corresponding values from the current fact level. This technique of "'on the fly"' instantiation allows avoiding the flood of numeric values and can be used for continuous and discrete numeric values. Only the variables that are parameters of external functions and the right hand side numerical variables of expressions are instantiated from the current fact level. All these on the fly instantiated variables are added implicitly to the action preconditions by direct edges. These edges to implicit preconditions will allow us to follow the trace of the functions application within the planning graph during the extraction process. We run through the planning graph during the extraction of a plan without making difference between implicit and explicit preconditions. This technique allows us to use black box functions because we are only concerned by the parameters and the returned value of a function to follow its trace through edges during the planning process. After all actions are instantiated propositionally and numerically at the current level, the application of these actions can take place.

The application of actions at the current action level will lead to the creation of the next fact level. Positive propositional effects in eff_P of an applicable action a are pointed as added by a to the next fact level and negative effects are pointed as deleted by a from the current fact level using edges. For each numerical variable which is assigned to an external function or to an expression at the numerical effects in eff_N, the old values of these variables are pointed as deleted by a from the current fact level and the new values returned by functions or expressions are pointed as added by a to the next current level. This organization of numerical effects will allow a uniform representation for all types of values and thus, the run through the graph will not be affected by the function results. From now on, the next fact level becomes the current fact level and the loop continue until the goal conditions are found at this current fact level or until a sufficient condition indicates that the problem has no solution.

6 CONCLUSION

In this paper we proposed an extension to the planning graph construction algorithm in order to allow it supporting the application of functions written in a given programming language. This extension allows the application of functions in a planning graph irrespective of their contents. As the planning graph is build in a forward pass the parameters and the results of the functions can be registered consequently during the graph construction process. During the extraction of a plan in a backward pass the algorithm is only concerned by the graph nodes from which it finds a solution by following the direct edges of the graph. We have proposed a method to instantiate numeric variables dynamically, in a way that the instantiation of these variables is done step by step during the graph expansion process. This technique resolves the problem of the flood that can be generated by the numeric domains and allows handling continuous and discrete numerical domains during a planning task.

REFERENCES

A., G. and D., L. (2005). Plan constraints and preferences for PDDL3. Technical Report Technical report, R.T. 2005-08-07, Dept. of Electronics for Automation, U. of Brescia, Brescia, Italy.

BAK M., P. N. and O., R. (2000). Path following mobile robot in the presence of velocity constraints. Technical report, Kongens Lyngby, Denmark : Technical University of Denmark.

Blum, A. L. and Furst, M. L. (1995). Fast planning through planning graph analysis. *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI95)*, pages 1636–1642.

D., S. and D., W. (1999). Temporal planning with mutual exclusion reasoning. *16th International Joint Conference on Artificial Intelligence (IJCAI99)*.

Do, M. B. and Kambhampati, S. (2001). Sapa: A domain-independent heuristic metric temporel planner. *In Proceedings of the European Conference on Planning (ECP 2001)*.

Geffner, H. (1999). Functional strips: A more flexible language for planning and problem solving. *Logic-based AI Workshop, Washington D.C.*

Hoffman, J. (2001). FF: The fast-forward planning system. *AI Magazine*, 22:57 – 62.

Hoffmann, J. (2002). Extending FF to numerical state variables. *In Proceedings of the 15th European Conference on Artificial Intelligence, Lyon, France*.

Schmid, U., Müller, M., and Wysotzki, F. (2000). Integrating function application in state-based planning. *Submitted manuscript*.

Smith, D. and Weld, D. (1999). Temporal planning with mutual exclusion reasoning. *In Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-99)*.

MULTICRITERIAL DECISION-MAKING IN ROBOT SOCCER STRATEGIES

Petr Tučník
University of Hradec Králové
Náměstí svobody 331, 500 01, Hradec Králové
petr.tucnik@uhk.cz

Jan Kožaný and Vilém Srovnal
VŠB - Technical University of Ostrava
17. listopadu 15, 708 33, Ostrava-Poruba
{jan.kozany,vilem.srovnal}@vsb.cz

Keywords: Multicriterial decision-making, robot soccer strategy.

Abstract: The principle of multicriterial decision-making is used for the purpose of autonomous control of both individual agent and the multiagent team as a whole. This approach to the realization of control mechanism is non-standard and experimental and the robot soccer game was chosen as a testing ground for this control method. It provides an area for further study and research and some of the details of its design will be presented in this paper.

1 INTRODUCTION

A great deal of scientific effort is aimed at the multiagent systems today. Also, the area of multicriterial decision-making (or decision-support) is well developed. Nevertheless, the combination of both principles mentioned above still stands aside of the scientific interest focus and provides a space for further study.

The need to follow the restrictions as well as semantic meanings of general definitions provided by both principles is the reason why it is necessary to re-define some notions from the point of view of multicriterial decision-making (MDM) in multiagent systems (MAS). Also, the principle changes of both little or big importance have to be made accordingly. It is important to remember the fact, that there are restrictions and constraints, i.e. above all, the need of numerical representation of all the facts relevant for the decision-making (DM) process.

The MDM in MAS shows many features which make such approach at least very interesting. It is a method that is capable of providing an autonomous control to the MAS as a whole or to the individual agent. We have focused our research of this DM system on the problem of the robot soccer game. The game is fast, quickly evolving and represents a dynamically changing environment, where it is impossible to follow long-term plans. We have to make decisions based on the limited set of attributes (most of them have to be computed from the processed images), while there is a theoretically infinite amount of solutions and a minimal number of possible actuator actions.

2 THE THEORETICAL PART

We introduced the basics of MDM terms in (Tučník et al., 2006). In (Ramík, 1999), (Fiala et al., 1997), the wide scale of methods of multi-criteria decision-making may befound. The Fig. 1 shows the steps of the MDM process.

After initialization and goal acquisition in the phase 0, the agent tries to refresh its environmental data by its sensors – this is the phase 1. The agent's environment is described by attributies. Each attribute is expressed numerically and represents measure of presence of given characteristic in the environment. The attribute value fits into bordered interval the endpoints (*upper limit* and *lower limit*) of which

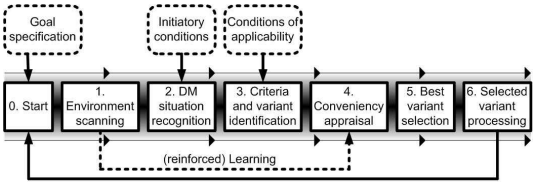


Figure 1: The MDM process.

are specified by the sensor sensitivity and range. During the signal processing, the standard normalization formula

$$\text{normalized\_value} = \frac{\text{attribute\_value} - k}{l - k} \quad (1)$$

is used. The attribute value is normalized to the interval of $\langle 0; 1\rangle$ and variables k, l are expressing the lower and upper limit of the sensor scanning range.

The set of initiatory conditions is used for the decision-making situation recognition in the phase 2. It has to provide the agent with the ability to react to unexpected changes in the environment during its target pursue. Also, it must allow the agent to change its goal through the decision-making process pertinently, e.g. if the prior goal is accomplished or unattainable. In the phase 3, the conditions of applicability characterize the boundary that expresses the relevance of using the variant in the decision-making process. The most important step is the phase 4, where the convenience appraisal of applicable variants is performed. This step plays the key part in the decision-making procedure. The following formula is used:

$$\text{conv} = \sum_i w(\text{inv}(\text{norm}(a_i^v))), i = 1, \ldots, z, \quad (2)$$

where v stands for the total number of variants and z is total number of attribute values that are needed for computation of convenience value.

The convenience value for the each of assorted variants is obtained. Attributes a_i^v stand for presumptive values of universum-configuration and the *norm* function normalizes the value of the attribute and is mentioned above (1). The function *inv* is important, as it represents reversed value of difference between real attribute value and ideal attribute value:

$$\text{inv}(\text{current\_val}) = 1 - |\text{ideal\_val} - \text{current\_val}|. \quad (3)$$

The optimal variant remains constantly defined by m-nary vector of attributes, where $m \leq z$, for the each of the decision-making situations and attributes a_i^v differ for each variant other than the optimal variant. There is a final number of activities that the agent is able to perform. As the inverse values of difference between the real and ideal variant are used, in the most ideal case, the convenience value will be equal to 1, and in the worst case it will be close to 0. $\text{Im}(\text{inv}) = (0; 1\rangle$. The lower open boundary of the interval is useful, because troubles related to computation with zero (dividing operations) may be avoided.

The function w assigns the importance value (weight) to the each attribute. The machine learning is realized by proper modifications of the weight function. Importance of attributes differs in accordance with the actual state of the agent. E.g. energetically economical solution would be preferred when the battery is low, fast solution is preferred when there is a little time left, etc. Precise definitions of weight functions are presented in (Ramík, 1999), (Fiala et al., 1997). In the phase 5, the variant with the highest convenience value is selected and its realization is carried out in the phase 6. During processing of the selected solution, the agent is scanning the environment and if the decision-making situation is recognized, the whole sequence is repeated. The evaluation function (e.g. reinforced learning functions examples in (Kubík, 2000), (Pfeifer and Scheier, 1999), (Weiss, 1999)) provides the feedback and supports the best variant selection, as it helps the agent in its goal pursue. Based on the scanned environmental data, modifications of the function w are made during the learning process.

3 THE PRACTICAL PART

As it was said above the robot soccer game is a strongly dynamical environment (Kim et al., 2004). Any attempt to follow a long-term plan will very probably result in a failure. This is a reason why there are many solutions of this problem founded on the agent-reactive basis. But such reactive approach lacks potential to develop or follow any strategy apart from the one implemented in its reactive behavior.

On the other hand, the MDM principle, in general, provides a large variety of solutions and the strategy may be formed by the modifications of the weight coefficients value. Such modifications change behavior of the team as a whole and/or its members individually.

Simple reactions do not provide sufficient effort potential, or, in other words, the efficient goal pursue. There is a need of a quick, yet more complicated, actions, that would allow us to build up a strategy. Also,

there is an important fact that the efficient strategy must not omit the agent cooperation.

3.1 Centralized vs. Autonomous Control

As there are two possible points of view – individual (agent) and global (team) - there are also two corresponding goals for the each agent. The global goals represent target state of the game for the team as a whole and pursue of global goal is superior to the individual goal of each agent. Therefore, the aim of individual goal has to correspond with the global goal. However, the emphasis on the autonomous behavior allows the agent to temporarily damage the team's effort of the global goal pursue in order to improve its position significantly in the next iteration(s). The autonomous aspect of the multiagent approach is therefore used to avoid local maximum lock-down. These ideas underline our approach to the solution of this problem.

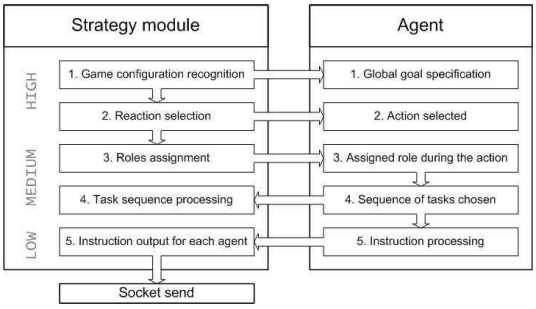


Figure 2: The scheme of communication between the strategy module and agents (the agent on the right stands for any agent of our team). At the beginning the game configuration is recognized and goal specified for each agent. In the second phase is chosen the most appropriate reaction in the form of action the agent has to perform. It is also assigned a role in the scope of this action. Then the agent itself forms a sequence of small tasks, which leads it towards advance in its goal pursue. These tasks are also processed in the strategy module, because of possible conflict check. As the last step, the set of simple instructions (i.e. practically HW instructions) is generated accordingly. These instructions represent the final output of the strategy module.

In order to gain maximum profit from both global (centralized) and individual (autonomous) approaches, the incremental degradation of centralization towards the autonomy was chosen during the strategy selection process. This means assignment of more individual instructions at the end of the strategy forming process and, on the contrary, the global team role assignment at its beginning. Fig. 2 shows the sequence of the decision output assigned to the agent.

There are three levels of the strategy module corresponding with three levels of decisions made. The high level strategy is responsible for the game configuration assessment and selection of the most appropriate actions. It is also a level, where the most centralized evaluations and global decisions are made.

The second layer of the medium level strategy is where the role and task assignment operations are performed. The notion of a "task" refers to the quite simple sequence of orders, for example: *GO TO POSITION X, Y, ANGLE 60°*.

Such simple tasks are transformed into instructions in the third (and final) low level strategy layer. These instructions are hardware-level commands and are used to set the velocity of the left and right wheel of the robot, separately. Because these instructions represents formatted output of the strategy module, they require no more of our further attention, as it is no aim of this paper to discuss such issues.

3.2 High Level Strategy

The purpose of this part of the strategy module is to provide a game configuration analysis and to perform a multicriterial selection of an appropriate reaction to the given state (of the game). There is an infinite number of configurations. Therefore, limiting, simplifying supportive elements have been implemented.

We will mention just one of these simplifying tools. The first and most important supportive element of the DM process is a ball-possession attribute. As the ball is the most important item in the game, it also plays a crucial role in the strategy selection process. Whoever controls the ball controls the game. Accordingly to the ball-possession, the set of all possible strategies is divided into three subsets. These subsets are:

- offensive strategies (our team have the ball),
- defensive strategies (opponent has the ball) and
- conflict strategies (no one has the ball or the possession is controversial (both teams have players very close tothe ball)).

Each item of these subsets has attached a certain number of actions to it. These actions represent different ways of superior (global) goal pursue.

In the MDM system, the convenience value must be obtained (from the formula (2)) for each of the possible solutions. This procedure is called multicriterial decision making process (MDMP) in the following text.

Because we need enough information to make a qualified decision (see (Fiala et al., 1997), (Ramík, 1999)), the sufficient amount of attributes has to be present. But there is a limited amount of data from the video recognition module. This is the reason, why it is necessary to perform additional attribute computing and gain more information needed.

The notion of the game configuration was used before. It refers, in the first place, to the position and angle of orientation of the each agent (robot) and to the position of a ball in the game. These are basics we can use to form a prediction of state of the game in next iterations. The prediction is based on the game configuration data (GCD) of the actual iteration and a few iterations back. However, it is vain to try to predict too far into the future, as the game is developing and the environment changing quickly.

Such approach to information extraction ensures enough data for GCD processing. The ACTION selected afterwards represents reaction of the multiagent system to the present situation. Such "ACTION" is the basic concept for the behavior scheme and few steps in the medium layer of the strategy module have to follow until the final output (command) is generated.

3.3 Medium Level Strategy

The function of this layer of the strategy module is to assign the role to the agent and to ensure, that the possible collisions of plans between team-mates will be avoided. The high level of the strategy module has selected an appropriate action to the given state of the game. Further steps during the DM process have to be taken.

One of the most important parts of the ACTION structure is the ROLE. The ROLE stands for function the agent has to guarantee during the subsequent performance of chosen activity. The ROLE may be either leader or support. For the leader role, the most convenient agent is assigned. The convenience value is calculated using the formula (2), and this calculation is preferentially based on its position and angle of orientation and additional supportive attributes, such as prediction of positions and movement trajectories. The support roles are assigned in a similar manner.

The most important are actions of the leading agent with the ball. All other activities are inferior to it and all agents required have to cooperate. However, in the opposite case, there may be a spare potential of non-employed agents and its utilization should be arranged.

As the back-up solution, there is also present a standard behavior algorithm. Such behavior is applied when there are no other actions possible or suitable for given situation, or not enough agents available for execution.

As it is shown in the Fig. 2, the selection of the sequence of tasks is the next step. Every action may be divided into atomic tasks. This decomposition forms desired sequence and check operations are performed to avoid collisions and other errors or mistakes. The possible necessary adaptations are made in a centralized manner and all agents proceed to the final low-level layer of the strategy module.

4 CONCLUSION AND FUTURE WORK

The implementation of the MDM-based control mechanism is an experimental matter. The robot soccer game provides the testing and proving ground for this approach. Further tests have to be performed as well as adaptation and optimization work. However, the principle shows a promising potential and it is able to function as a control mechanism and decision-making tool.

ACKNOWLEDGEMENTS

The work and the contribution were supported by the project from Grant Agency of Czech Academy of Science - Strategic control of the systems with multiagents, No. 1ET101940418 (2004-2008) and by the project from the Czech Republic Foundation - AMI-MADES, No. 402/06/1325.

REFERENCES

Fiala, P., Jablonský, J., and Manas, J. (1997). *Vícekriteriální rozhodování*. VŠE Praha.

Kim, J., Kim, D., Kim, Y., and Seow, K. (2004). *Soccer Robotics (Springer Tracts in Advanced Robotics)*. Springer-Verlag.

Kubík, A. (2000). *Agenty a multiagentové systémy*. Silesian University, Opava.

Pfeifer, R. and Scheier, C. (1999). *Understanding Intelligence*. Cambridge, Massechusetts.

Ramík, J. (1999). *Vícekriteriální rozhodování – Analytický hierarchický proces (AHP)*. Silesian University.

Tučník, P., Kožaný, J., and Srovnal, V. (2006). Multicriterial decision-making in multiagent systems. *Lecture Notes in Computer Science*, pages 711 – 718.

Weiss, G. (1999). *Multiagent Systems – A Modern Approach to Distributed Artificial Intelligence*. Cambridge, Massechusetts.

COGNITIVE APPROACH TO PROBLEM SOLVING OF SOCIAL AND ECONOMIC OBJECT DEVELOPMENT

Z. Avdeeva, S. Kovriga and D. Makarenko
Institute of Control Sciences of Russian Academy of Sciences, Profsouznaya st., Moscow, 117997, Russia
max@ipu.ru, maxi@ipu.ru, maxim@ipu.ru

Keywords: Strategic management, social-economic object, problem-solving, decision support, semi-structured problems, cognitive map, cognitive modeling.

Abstract: The basic technique of problem-solving is structurization of knowledge about object and its environment and construction of a cognitive model. The technique includes monitoring of dynamics of factors of the model (their tendencies), analysis of the model structure with the use of SWOT-approach, and modeling that permits to determine and solve semi-structured problems. The technique allows supporting of a vital control task that consists in goal setting of socio-economic object development, as far as solution of discovered problems turns into the system development control task. The application of technique is useful when designing a strategy of development of social and economic objects.

1 INTRODUCTION

Traditional methods of decision theory concentrate attention on processes of search of optimal solution amongst the fixed set of alternatives in order to achieve accurately set goals. Problem identification, goal setting, and alternatives generating issues, as a rule, stay apart.

In real situations of social and economic object (enterprises, cities, regions, states, etc.) control there often arises a task, which consists in the analysis of a situation in order to reveal a problem and the reasons of it's appearance.

Cognitive approach to SEO modelling is directed to development of formal models and methods that support the intellectual process of problem solving owing to taking into account the cognitive abilities (perception, representation, cognition, understanding, explanation) of control agents.

The Institute of Control Sciences of the Russian Academy of Sciences has developed the technique of problem solving of SEO development on the basis of cognitive approach to modeling.

Cognitive modelling is a research of SEO functioning and development by means of creation of a situation model on the basis of a cognitive map. In the model a cognitive map reflects the subjective representations of a problem or a situation, connected with SEO functioning and development.

2 THE TECHNIQUE OF PROBLEM SOLVING OF SEO DEVELOPMENT

2.1 Construction of Model of SEO Development

The model construction is based on the cognitive structurization of knowledge about object and its environment.

The purpose of structuring is revealing of the most essential factors describe a “boundary” layer of interaction of object and external environment, and establishment of qualitative relationships of cause and effect between them.

The cognitive structurization is finished by formalization of singled out knowledge that consists in generalization of essential information into a system of basic factors.

Analysis of a graph model of a situation associated with a cognitive map allows to reveal the structural properties of a situation. The basis of the model is a weighed digraph G =(X, A), where X is a set of nodes that biuniquely corresponds to the set of basic factors, A is a set of arcs reflecting the fact of direct influence of factors. Each arc connecting some factor xi with some factor x_j has the weight a_{ij} which sign depicts the sign of influence of the factor xi on the factor x_j, and the absolute value of a_{ij}

depicts the strength of the influence. Thus, the cognitive map can be examined as a connectivity matrix A_g of the graph G.

When constructing a cognitive map of SEO development set of basic factors, X, is grouped in blocks relevant to external environment, X^{ext}, and an internal environment SEO, $X^{int} = X \backslash X^{ext}$. Besides determination of factors and influence between them the vector of initial factor trends, $X^{ext}(0) \cup X^{int}(0)$, is established.

The dynamic model is constructed to obtain a new knowledge of structure and dynamics of situations under research. On the basis of that model the one carries out scenario research with use of methods of computer modelling of self-development and controlled development of SEO (Maximov, 2001; Maximov and Kornoushenko, 2001; Avdeeva, et al., 2003; Makarenko, et al., 2004).

Behavior of factors in time is described by dynamic linear model

$$x_i(t+1) = x_i(t) + \sum_{j \in I_i} b_{ij} \left(x_j(t) - x_j(t-1) \right), \qquad (1)$$

where $x_i(t+1)$ and $x_i(t)$ – value of factor i in t+1 and t time points accordingly, i=1, 2, …, N; $x_j(t) - x_j(t-1) = \Delta x_j(t)$ – increment of factor x_j in t time point, describing the rate of change (tendency) of x_j; b_{ij} – strength of influence of factor x_j on factor x_i, corresponding to matrix B element ($B = A_g^{'}$); I_i –set of numbers of factors directly influencing the factor x_i.

The input parameters are the initial rates of change of factors (initial tendencies), that describe a situation's history. Actually, values of factors are not fixed, i.e. dynamics of change of a situation is being simulated on the basis of factors' increments.

2.2 SWOT-Analysis on the Basis of Analysis of the Model Structure

Generally, SWOT-analysis is expert determination of strength and weaknesses of SEO, opportunities and threats of its environment, and estimation of their interaction. Results of SWOT-analysis are represented as a matrix “Window of opportunities”.

The mathematical procedure of generation of matrix “window of opportunities” on the basis of analysis of structural features of SEO development cognitive map has been worked out. Thus, there is no necessity in regular attraction of experts with all accompanying procedures.

The essence of the procedure of SWOT-analysis is the following.

Analyzing a situation of SEO development, it is possible to put forward various hypotheses about desirable dynamics of any factor of the model. So, the parameter “attitude to factor dynamics” (AFD) is brought in for each factor of the model. If dynamics of a factor is positive (negative) AFD is equal 1 (–1). If it is difficult to evaluate the factor dynamics its AFD is equal 0. The set of AFD vector on some set of model factors reflects desirable change of a situation in SEO.

Let's designate through $R^{ext}(X^{ext})$, $R^{int}(X^{int})$ vectors of AFD of factors of the external and internal environment, where $X^{ext} \cup X^{int} = X$; $X^{ext}(0)$, $X^{int}(0)$ – vectors of initial trends accordingly.

While the situation evolves each factor is being influenced not only by “neighbouring” factors, but also by more “distant” ones and these indirect influences are transferred through chains of the appropriate factors and graph arcs that connect them. Set of influences both direct, and indirect to which each factor in a situation is subject, is described with the use of concept of transitive closure of a cognitive map of the situation.

To determine the transitive closure it will suffice to consider N terms in a power series of matrix B, where N - the order of matrix B, i.e. number of basic factors in a cognitive map of a situation. Then the transitive closure of matrix B is estimated by matrix:

$$Q = E_N + B + B^2 + \ldots + B^N \cong (E_N - B)^{-1} \qquad (2)$$

When constructing a matrix “Window of opportunities” opportunities and threats of the environment, strengths and weaknesses of SEO are determined on the basis of observation of the dynamics of model factors and estimations of their integrated influence on desirable dynamics of factors $R^{ext}(X^{ext})$ и $R^{int}(X^{int})$. The significance of strengths and weaknesses of SEO is determined as well.

Let us introduce the basic definitions.

Definition 1. If the initial trend of the internal environment factor, $x_i^{int}(0)$, is negative, i.e. does not correspond to a desirable direction of change (AFD), the given factor is regarded as a weakness of functioning and development of SEO, otherwise (the trend is favourable) - as a strength of object. The weaknesses determine internal threatening trends to SEO development, and strengths - internal favourable trends.

Using a terminology of SWOT-analysis, we shall designate X^{st} – a subset of factors–strengths of SEO, X^{w} – a subset of the factor–weaknesses of SEO, $X^{st} \cup X^{w} \subset X^{int}$.

Definition 2. The initial factor trend from X influences positively on desirable dynamics of the factor from X if the following is fair

$$\text{sign}(x_i(0)q_{ij}) = r_j(x_j),$$

where q_{ij} – (i,j) element of a transitive closure matrix Q, which determines integrated influence of i factor on the j factor; $q_{ij}=0$ if x_i does not influence x_j.

If AFD of some factors are given equal to zero ($r_j(x_j) = 0$) such factors are excluded from the analysis (integrated influences on them of other factors are not taken into account).

The following definitions follow from definition 2.

Definition 3. The factor of an environment x_i^{ext} is neutral for X^{int}, if the initial trend of this factor does not influence ($q_{ij}=0$) the desirable dynamics of all factors of the internal environment of SEO, X^{int}.

Definition 4. The factor of environment x_i^{ext} characterizes the opportunity for SEO development if the factor is not neutral and its initial trends does not negatively influence (through the appropriate integrated influences) the desirable dynamics of all factors of internal environment of SEO, $R^{int}(X^{int})$. In other words the initial trend of factor x_i^{ext} promotes SEO development in a desirable direction.

Definition 5. The trend of the environment factor x_i^{ext} threatens the SEO development, if it negatively influences (through the appropriate integrated influences) desirable dynamics even of one factor of internal environment $x_j^{int} \in X^{int}$.

Using a terminology of SWOT-analysis, we shall designate X^{op} – a subset of factors–opportunities for SEO development, X^{th} – a subset of factors–threats to SEO development, $X^{op} \cup X^{th} \subset X^{ext}$.

Definition 6. The internal environment factor, x_i^{int} ,is neutral for X^{ext}, if the initial trend of the factor does not influence ($q_{ij}=0$) the desirable dynamics of the environment factors X^{ext}.

Definition 7. The internal environment factor, x_i^{int}, promotes strengthening of the opportunity of environment x_j^{op} if x_i^{int} is not neutral and its initial trend favourably influences (through the appropriate integrated influence) the desirable dynamics of the factor x_j^{op}. Otherwise x_i^{int} promotes decrease of the opportunity of environment.

Definition 8. The internal environment factor x_i^{int} promotes parrying of threats of environment x_j^{th} if x_i^{int} is not neutral and its initial trend favorably influences (through the appropriate integrated influence) the desirable dynamics of the factor x_j^{th}. Otherwise x_i^{int} promotes strengthening of threats of environment.

On the basis of definitions 1-8 SWOT-analysis comes to the following stages:

1. Building of cognitive map of SEO development with extraction of external X^{ext} and internal blocks of factors. Vector of initial trends of factors $X^{ext}(0)$ and $X^{int}(0)$ is set.
2. AFD for each factor $R^{ext}(X^{ext})$, $R^{int}(X^{int})$ is set.
3. Strengths and weaknesses for each object (X^{st} and X^{w}) are found on the basis of the vector $X^{int}(0)$.
4. Matrix of transitive closure Q (2) is used to build

– matrix “Window of opportunities ext-int” on the basis of which opportunities X^{op} and threats X^{th} of environment and their importance (how great is their influence on factors of internal environment) are determined, $X^{op} \cup X^{th} \subset X^{ext}$;

– matrix “Window of opportunities int-ext” with the purpose of determination of internal opportunities of SEO that can neutralize the threats of environment X^{th}, and also the problems connected with possible negative influence of SEO on environment X^{ext};

– matrix “Window of opportunities op-th”. The analysis of interferences between opportunities and threats allows to reveal opportunities which promote parrying of threats;

– matrix “Window of opportunities st-w” for revealing the latent internal opportunities allowing to remove weaknesses of SEO due to advantages.

As a result of the analysis all factors are being grouped into the following classes: S (Strengths), W (Weaknesses), O (Opportunities), T (Threats). Factors inside of each class are being ranked according to the force of their favourable (unfavourable) influence on the factors of another class. This procedure lets us estimate the importance of strengths and weaknesses, opportunities and threats for SEO development.

2.3 Structure and Goal Analysis of SEO Development

When setting the goals of a SEO development a decision maker doesn't always manage to trace if the goals he has set are inconsistent, i.e. reaching of a goal will prevent from reaching of another one. Inconsistency of goals can also be influenced by the chosen ways of their reaching.

Thus, it is very important to reveal the contradictions already at the stage of goal setting.

The technique of the structure and goal analysis of SEO development. allows to determine integrated (direct and all possible indirect) influences of one factor on the other and due to it to reveal

inconsistencies between goal and control factors. The structure and goal analysis also allows to determine the most effective controls.

Thus, the structural analysis of cognitive model of a situation development under control consists of the following stages:

Stage 1 - analysis of goals (coordinates of a vector of goals) on mutual consistency in order to answer the question "whether the vector of goals (fixed or unfixed) is inconsistent, i.e. whether the reaching of any of goals (coordinates in a vector of the given goals) will prevent from reaching of other goals?"

When the consistent vector of goals is formed the desirable integrated change of any of goal factors will not result in undesirable integrated change of other goal factors in a vector of goals.

Stage 2 - check of a consistency of the set of control factors with the given vector of goals, i.e. whether the change of the value of any control factor (with the help of the appropriate control) will promote reaching of some goals in a vector of goals and at the same time prevent from reaching of other goals of a vector of goals.

Stage 3 - estimation of efficiency of influence of control factors on all coordinates of the vector of goals. Such estimation is useful when choosing the most effective control factors the changes of which with the help of the selected control actions will provide the purposeful development of a situation.

Formally, the parameter of efficiency is determined as absolute value of the sum of coefficients of influence of the given control factor on the goal factors multiplied by AFD of the goal factors.

2.4 Scenario Modelling

Scenario research (Maximov, 2001; Maximov and Kornoushenko, 2001; Avdeeva, et al., 2003; Makarenko, et al., 2004) allows to develop various variants of SEO development in view of arising trends (favourable and negative) of environment.

Scenario consists of a set of factor trends describing a situation at the present moment, desirable goals, a set of activities that are used upon the running of a situation, and system of observable parameters (factors) illustrating behaviour of processes.

Self-development of a situation represents evolution of an initial condition of a situation in SEO. Results of such modelling answer the question: what can take place of the existing trends of SEO development are preserved?

Controlled development of a situation is determined by the chosen goal of control.

On the basis of results of SWOT- and Structure and goal analysis we form the set of vectors of goals of SEO and vectors of controls that reflect possible strategy of secure SEO development.

The combination of the stages described above enables one to diagnose and define problems in SEO development and find ways of its solution and form well-founded goals and strategy of SEO development.

3 CONCLUSIONS

The result of application of the technique is the project of strategy of SEO development in the form of goals and main ways of its reaching. Goal-setting issues from forecasting and analysis of problems. Cognitive model, which is used in the basis of the technique, gives an opportunity to use monitoring data and quickly correct models with the purpose to analyzing new problems.

REFERENCES

Avdeeva, Z., Kovriga, S., Makarenko, D., and Maximov, V., 2003. Goal Setting and Structure and Goal Analysis of Complex Systems and Situations. In *ASBoHS'03*, *8th IFAC Symposium on Automated Systems Based on Human Skill and Knowledge.* Göteborg: IFAC.

Makarenko, D., Avdeeva, Z., and Maximov, V., 2004. Cognitive Approach to Control of Socio-Economic Systems Security. In *SMC'04, Proceedings of the IEEE International Conference "Systems, Men & Cybernetics"*. Hague: IEEE.

Maximov, V., Kornoushenko, E., 2001. Analytical Basics of Construction the Graph and Computer Models for Complicated Situations. In *INCOM'01, Proceedings of the 10th IFAC Symposium on Information Control Problems in Manufacturing.* Vienna: IFAC.

Maximov, V., 2001. Cognitive Analysis and Situation Modelling. In *SWIIS'01. Proceedings of the 8th IFAC Conference on "Social Stability: The Challenge of Technology Development"*. Vienna: IFAC.

TAKAGI-SUGENO MULTIPLE-MODEL CONTROLLER FOR A CONTINUOUS BAKING YEAST FERMENTATION PROCESS

Enrique Herrera[1], Bernardino Castillo
Centro de investigación y de Estudios Avanzados del I.P.N., Unidad Guadalajara, Av. Científica 1145, Colonia el bajío C.P. 45010, Zapopan, México
toledo@gdl.cinvestav.mx, eherrera@ciatej.net.mx

Jesús Ramírez
Centro de Investigación y Asistencia en Tecnología y Diseño del Estado de Jalisco A.C., Av. Normalistas 800 C.P. 44270, Guadalajara, México
jramirez@ciatej.net.mx

Eugénio C. Ferreira
IBB-Institute for Biotechnology and Bioengineering, Centre of Biological Engineering, Universidade do Minho 4710-057, Braga, Portugal
ecferreira@deb.uminho.pt

Keywords: Nonlinear system, Fuzzy model, Fuzzy controller, Fermentation process.

Abstract: The purpose of this work is to design a fuzzy integral controller to force the switching of a bioprocess between two different metabolic states. A continuous baking yeast culture is divided in two sub-models: a respiro-fermentative with ethanol production and a respirative with ethanol consumption. The switching between both different metabolic states is achieved by means of tracking a reference substrate signal. A substrate fuzzy integral controller model using sector nonlinearity was built for both nonlinear models.

1 INTRODUCTION

Control applications in bioprocesses have increased in the last decades due to the fast advances on computer and electronic technology. An adequate control of fermentation processes allows reducing production costs and increases the yield, while at the same time achieving the quality of the desired product (Yamuna and Ramachandra 1990).

In the case where the nonlinear model of the process is known, a fuzzy system may be used. A first approach can be done using the Takagi-Sugeno (TS) fuzzy model, (Takagi and Sugeno, 1985), where the consequent part of the fuzzy rules are replaced by linear systems. This can be attained, for example, using the method of sector nonlinearities which allows constructing an exact fuzzy model from the original nonlinear system by means of linear subsystems (Tanaka and Wang, 2001). From this exact model a controller may be designed based on the linear subsystems.

Along this line of reasoning, in this work a fuzzy integral controller based on sector nonlinearities is proposed and applied to a continuous baker's yeast process. An interesting feature of this model is the splitting in two different partial models: a respiro-fermentative (RF) model with ethanol production and the respirative (R) model with ethanol consumption. The fuzzy integral controller is used to force the switching of a bioprocess between both different metabolic states by means of tracking a reference substrate signal.

[1]On Ph. D. studies leave from the Centro de Investigación y Asistencia en Tecnología y Diseño del Estado de Jalisco.

2 FUZZY MODELS BASICS

2.1 Takagi-Sugeno Fuzzy Model

The Takagi-Sugeno fuzzy models are used to represent nonlinear dynamics by means of a set of IF-THEN rules. The consequent parts of the rules are local linear systems. The *i*th rule of a continuous fuzzy model has the following form:

$$\begin{aligned} &IF\ z_1(t)\ is\ M_1^i\ and...and\ z_p(t)\ is\ M_p^i \\ &THEN \begin{cases} \dot{x}(t) = A_i x(t) + B_i u(t) \\ y(t) = C_i x(t) \end{cases} \quad i = 1,...,r, \end{aligned} \tag{1}$$

where M_j^i is a fuzzy set and *r* is the number of rules in the fuzzy model; $x(t) \in R^n$ is the state vector, $u(t) \in R^m$ is the input vector, $y(t) \in R^q$ is the output vector, $A_i \in R^{nxn}$, $B_i \in R^{nxm}$, $y\ C_i \in R^{qxn}$ are suitable matrices and $z(t)=[z_1(t),\ldots,z_p(t)]$ is a known vector of premise variables which may depend partially on the state $x(t)$. Given a pair of $(x(t), u(t))$ and a product inference engine the aggregate TS fuzzy model can be inferred as:

$$\begin{aligned} \dot{x}(t) &= \sum_{i=1}^{r} h_i(z(t)) \{ A_i x(t) + B_i u(t) \}, \\ y(t) &= \sum_{i=1}^{r} h_i(z(t)) C_i x(t), \end{aligned} \tag{2}$$

where

$$h_i(z(t)) = \prod_{j=1}^{p} M_j^i(z_j(t)) \Big/ \sum_{i=1}^{r} \left(\prod_{j=1}^{p} M_j^i(z_j(t)) \right),$$

for all *t*. The term $M_j^i(z_j(t))$ is the membership value of $z_j(t)$ in M_j^i. We have that $h_i(z(t)) \geq 0$ and $\sum_{i=1}^{r} h_i(z(t)) = 1$ for all *t* and i=1,…,r.

2.2 Parallel Distributed Compensator

The parallel distributed compensator (PDC) is used to design a fuzzy controller from a TS fuzzy model. Each control rule is designed from the corresponding rule of a TS model.

$$\begin{aligned} &IF\ z_1(t)\ is\ M_1^i\ and...and\ z_p(t)\ is\ M_p^i \\ &THEN \quad u(t) = -\sum_{i=1}^{r} h_i(z(t)) F_i x(t) \quad i = 1,...,r, \end{aligned} \tag{3}$$

where F_i is the controller gain for the *i*th subsystem, which makes Hurwitz the matrices A_i-B_iF_i.

2.3 Integral Control

Consider the linear system

$$\dot{\xi}(t) = A_\xi \xi(t) + B_\xi u_\xi(t) + y_R(t), \tag{4}$$

where $\dot{\xi}, \xi, A_\xi, B_\xi, u_\xi(t)$ and y_R are given by

$$\dot{\xi}(t) = \begin{bmatrix} \dot{x} \\ \dot{\sigma} \end{bmatrix}, \quad \xi(t) = \begin{bmatrix} x \\ \sigma \end{bmatrix}, \quad A_\xi = \begin{bmatrix} A & 0 \\ -C & 0 \end{bmatrix},$$
$$B_\xi = \begin{bmatrix} B \\ 0 \end{bmatrix}, \quad y_R(t) = \begin{bmatrix} 0 \\ y_R \end{bmatrix}, \quad u_\xi(t) = -Fx + k\sigma$$

where $\dot{\sigma} = e = y_R - y$ is a tracking error and $y_R(t)$ is a reference signal. It is desired to design a state feedback control such that $y(t) \to y_R(t)$ as $t \to \infty$ (Khalil, 1996). If the pair (A,B) is controllable and the following condition is achieved

$$rank \begin{bmatrix} A & B \\ -C & 0 \end{bmatrix} = n + p, \tag{5}$$

then it is possible to find a matrix *K* such that A_ξ-$B_\xi K$ is Hurwitz, assuring that $y(t) - y_R(t) \to 0$ as $t \to \infty$; where $K=[-F, k]$ and *k* must be nonsingular.

3 THE EXACT FUZZY CONTROLLER

A continuous baking yeast culture may be represented by the following nonlinear system $\dot{x}(t) = f_i(x(t)) + Bu(t) + d(x(t))$ where $f_i(x(t))$ describes a respiro-fermentative baking yeast partial model (RF) with ethanol production and a respirative baking yeast partial model (R) with ethanol consumption (Pormeleau, 1990). The RF partial model is described by

RF model

$$f_{RF} = \begin{bmatrix} \frac{x_4}{Ko+x_4} q_o^{\max} \left(Y_{O2} - Y_{\eta} \frac{Y_{O2}}{Y_O} \right) & Y_r q_s^{\max} \frac{x_1}{Ks+x_2} & 0 & 0 \\ \frac{x_4}{Ko+x_4} q_o^{\max} \left(-k_1 Y_{O2} + k_2 Y_{\eta} \frac{Y_{O2}}{Y_O} \right) & -k_2 Y_r q_s^{\max} \frac{x_1}{Ks+x_2} & 0 & 0 \\ -k_3 Y_{\eta} q_o^{\max} \frac{Y_{O2}}{Y_O} \frac{x_4}{Ko+x_4} & k_3 Y_r q_s^{\max} \frac{x_1}{Ks+x_2} & 0 & 0 \\ -k_5 Y_{O2} q_o^{\max} \frac{x_4}{Ko+x_4} & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \tag{6}$$

The R model can be divided in two sub models (Ferreira, 1995):

Rqe1 model

$$f_{Rqe1}=\begin{bmatrix} Y_e q_e^{\max} Ki \dfrac{x_3}{(Ke+x_3)(Ki+x_2)} & Y_O q_s^{\max} \dfrac{x_1}{Ks+x_2} & 0 & 0 \\ 0 & -k_1 Y_O q_s^{\max} \dfrac{x_1}{Ks+x_2} & 0 & 0 \\ -k_4 Y_e q_e^{\max} Ki \dfrac{x_3}{(Ke+x_3)(Ki+x_2)} & 0 & 0 & 0 \\ -k_6 Y_e q_e^{\max} Ki \dfrac{x3}{(Ke+x_3)(Ki+x_2)} & -k_5 Y_O q_s^{\max} \dfrac{x_1}{Ks+x_2} & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \quad (7)$$

Rqe2 model

$$f_{Rqe2}=\begin{bmatrix} Y_{O2} e q_o^{\max} \dfrac{x_4}{Ko+x_4} & \dfrac{x_1}{Ks+x_2} q_s^{\max} \left(Y_O - Y_{O2} e \dfrac{Y_O}{Y_{O2}} \right) & 0 & 0 \\ 0 & -k_1 Y_O q_s^{\max} \dfrac{x_1}{Ks+x_2} & 0 & 0 \\ -k_4 Y_{O2} e q_o^{\max} \dfrac{x_4}{Ko+x_4} & k_4 Y_{O2} e q_s^{\max} \dfrac{x_1}{Ks+x_2} \dfrac{Y_O}{Y_{O2}} & 0 & 0 \\ -k_6 Y_{O2} e q_o^{\max} \dfrac{x_4}{Ko+x_4} & \dfrac{x_1}{Ks+x_2} q_s^{\max} \left(-k_5 Y_O + k_6 Y_{O2} e \dfrac{Y_O}{Y_{O2}} \right) & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \quad (8)$$

The input B matrix for all the models is given by

$$B=[-x_1,\ -x_2+S_{in},\ -x_3,\ -x_4]^T \quad (9)$$

where x_1 is the biomass, x_2 is the substrate, x_3 is the ethanol, x_4 is the dissolved oxygen, S_{in} is the inlet substrate concentration, D is the dilution rate, $u(t)=D$. The yield coefficients k_1 *to* k_6 and the remaining parameters values are described in Ferreira, (1995). The oxygen transfer rate (OTR) is assumed to be a measurable and known perturbation, and thus d=[0 0 0 OTR]^T. Before designing a fuzzy controller an exact fuzzy model must be first built.

When the nonlinear dynamic model for the baking yeast is known, as well as all their parameters, a fuzzy exact model can be derived from the given nonlinear model. This requires a sector nonlinearity approach (Tanaka and Wang, 2001). From the models (6-9) the fuzzy exact model can be constructed. The premise variables for the RF partial model (6) and the input B matrix (9) are chosen as:

$$z_1(t)=\frac{x_4}{Ko+x_4} \qquad z_2(t)=\frac{x_1}{Ks+x_2}$$

$$z_{x_1}(t)=x_1,\ z_{x_2}(t)=x_2,\ z_{x_3}(t)=x_3,\ z_{x_4}(t)=x_4.$$

The membership functions can be obtained from $z(t)=\sum_{i=1}^{2} M_i(z(t))a_i$ where the following property $M_1(z(t))+M_2(z(t))=1$ must be accomplished (Tanaka and Wang , 2001). The linear subsystems A_{ijklmn}^{RF}, B_{ijklmn}^{RF} are derived from

$$A_{ijklmn}^{RF}=\begin{bmatrix} a_i q_o^{\max} \left(Y_{O2} - Y_r \dfrac{Y_{O2}}{Y_O} \right) & Y_r q_s^{\max} b_j & 0 & 0 \\ a_i q_o^{\max} \left(-k_1 Y_{O2} + k_2 Y_r \dfrac{Y_{O2}}{Y_O} \right) & -k_2 Y_r q_s^{\max} b_j & 0 & 0 \\ -k_3 a_i Y_r q_o^{\max} \dfrac{Y_{O2}}{Y_O} & k_3 Y_r q_s^{\max} b_j & 0 & 0 \\ -k_5 a_i Y_{O2} q_o^{\max} & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \quad (10)$$

$$B_{ijklmn}^{RF}=[-c_k,\ -d_l+S_{in},\ -e_m,\ -f_n]^T$$

$$i,j,k,l,m,n = 1,2.$$

where a_i, b_j, c_k, d_l, e_m, f_n are the maximum and minimum values of $z_1(t)$, $z_2(t)$, $z_{x1}(t)$, $z_{x2}(t)$, $z_{x3}(t)$ and $z_{x4}(t)$ respectively. The following ranges for $x_1(t)\in[0,\ 10]$, $x_2(t)\in[0,\ 1]$, $x_3(t)\in[0,\ 5]$ and $x_4(t)\in[0,\ 0.007]$ were assumed.

From the model (10) the substrate integral controller for the RF partial model can be designed using the following model:

$$A_j^{RF}=\begin{bmatrix} -k_2 Yr q_s^{\max} b_j & 0 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} x_2 \\ \sigma \end{bmatrix} \quad (11)$$

$$B_l^{RF}=[-d_l+S_{in},]^T$$

From models (10) and (11) the A_{ijklmn}^{RF}, B_{ijklmn}^{RF} matrices for the RF partial integral PDC can be written as

$$A_{ijklmn}^{RF}=\begin{bmatrix} a_i q_o^{\max} \left(Y_{O2} - Y_r \dfrac{Y_{O2}}{Y_O} \right) & Y_r q_s^{\max} b_j & 0 & 0 & 0 \\ a_i q_o^{\max} \left(-k_1 Y_{O2} + k_2 Y_r \dfrac{Y_{O2}}{Y_O} \right) & -k_2 Y_r q_s^{\max} b_j & 0 & 0 & 0 \\ -k_3 a_i Y_r q_o^{\max} \dfrac{Y_{O2}}{Y_O} & k_3 Y_r q_s^{\max} b_j & 0 & 0 & 0 \\ -k_5 a_i Y_{O2} q_o^{\max} & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ \sigma \end{bmatrix} \quad (12)$$

$$B_{ijklmn}^{RF}=[-c_k,\ -d_l+S_{in},\ -e_m,\ -f_n,\ 0]^T$$

$$i,j,k,l,m,n = 1,2.$$

A general fuzzy rule to infer all the fuzzy rules for the RF PDC can be stated as:

IF $z_1(t)$ is *"$M_{1i}(z_1(t))$" and* $z_2(t)$ is *"$M_{2j}(z_2(t))$" and* $zx1(t)$ is *"$M_{3k}(z_{x1}(t))$" and* $z_{x2}(t)$ is *"$M_{4l}(z_{x2}(t))$" and* $z_{x3}(t)$ is *"$M_{5m}(z_{x3}(t))$" and* $z_{x4}(t)$ is *"$M_{6n}(z_{x4}(t))$"*

$$THEN \quad u^{RF}(t)=-F_{ijklmn}x(t) \quad (13)$$

From (12) and using the notation given by (4) the aggregated fuzzy controller for the RF partial model turns to be

$$\dot{x}^{RF}(t)=\sum_{1}^{64}h_{\psi}(z(t))\left[\left\{A_{ijklmn}^{RF}-B_{ijklmn}^{RF}F_{ijklmn}\right\}\xi(t)+y_R(t)+d\right], \quad (14)$$

where

$$\begin{aligned}\psi &= n+2(m-1)+4(l-1)+8(k-1)\\ &\quad +16(j-1)+32(i-1),\\ h_{\psi}(z(t)) &= M_{1i}(z_1(t))M_{2j}(z_2(t))M_{3k}(z_{x_1}(t))\\ &\quad \times M_{4l}(z_{x_2}(t))M_{5m}(z_{x_3}(t))M_{6n}(z_{x_4}(t))\end{aligned} \quad (15)$$

It has to be noticed that x_1, x_3 and x_4 are not taken into account in the PDC design; this is because these states are not intended to be stabilized but to switch between the RF and R partial models. The fuzzy controller for the models Rqe1 and Rqe2 were constructed following the same procedure.

4 SIMULATION RESULTS

The application of the proposed controller scheme was simulated using MATLAB™. In order to force the switching between the RF and the R baking yeast partial models, the substrate fuzzy controller was forced to track a square reference signal, varied between 0.01 g/l and 0.07 g/l. S_{in} was set to 5 g/l. The behavior of the substrate fuzzy tracking controller as well as the biomass and ethanol behavior are shown in figure 1.

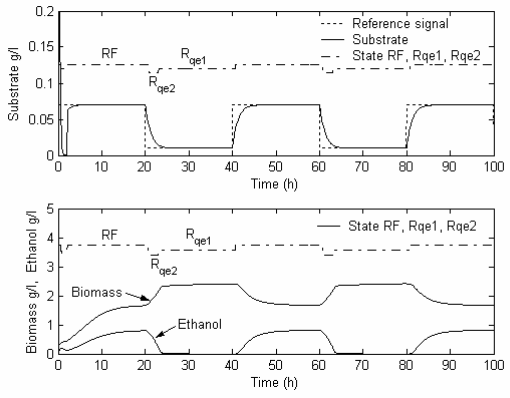


Figure 1: Fuzzy controller performance and biomass and ethanol behavior.

It can be notice that on the RF model, ethanol is produced, limiting the biomass growth; when the Rqe2 partial model is enable the ethanol is consumed promoting biomass growth, and on the Rqe1 state partial model the ethanol is consumed and the biomass is growing just due to the substrate uptake. To test the fuzzy integral controller performance, it was enabled when 2 hours of fermentation elapsed time was accomplished.

5 CONCLUSIONS

Based on the idea of splitting a continuous baking yeast model, a TS fuzzy model was proposed using the sector nonlinearities method, giving an exact representation of the original nonlinear plant. Moreover, a controller for each partial model was constructed. It is worth noting that the controller was capable to force the switching along the partial models. Therefore, the approach presented here may be considered a valid method to design a controller.

ACKNOWLEDGEMENTS

This paper has been supported by the Mexican Consejo Nacional de Ciencia y Tecnología (CONACyT), under grants 46538, 41148 and the Ph.D. Schoolarship 70662.

REFERENCES

Ferreira, E., 1995. *Identificação e controlo adaptivo de processos biotecnológicos*. Ph. D Thesis, Universidade do Porto, Portugal.

Khalil, H. 1996. *Nonlinear Systems*, second edition. Prentice Hall. United States of America.

Pormeleau, Y., 1990. *Modelisation et controle d'un procéde fed-batch de culture des levures á pain (saccharomyces cerevisiae)*. Ph. D. Thesis. Ecole Polytechnique de Montréal, Canada.

Takagi, T., Sugeno, M., 1985. *Fuzzy identification of systems and its applications to modelling and control*. IEEE Trans. Sys, Man Cyber, 15, 116-132.

Tanaka, K., Wang, H., 2001. *Fuzzy control systems design and analysis, a linear matrix inequality approach*. John Wiley & Sons, United States of America.

Yamuna, R., Ramachandra, R., 1999. *Control of fermenters - a review*. Bioprocess Engineering, 21, 77-88.

TOWARDS RELIABLE AUTOFOCUSING IN AUTOMATED MICROSCOPY

Silvie Luisa Brázdilová
Faculty of Informatics, Masaryk University, Botanická 68a, Brno, Czech Republic
luisa@mail.muni.cz

Keywords: Fluorescence microscopy, focus function, genetic programming.

Abstract: The results presented in this paper are twofold. First, autofocusing in automated microscopy is studied and evaluated with respect to biomedical samples whose images can have more than one in-focus planes. While the proposed procedure for finding the maximum of a focus function in a short time works satisfactorily, the focus function itself is identified as the weakest link of the whole process.
Second, an interesting property of functions used for genetic programming, and an algorithm for generating new individuals are introduced. Their usefulness and applicability are demonstrated on the problem of finding a new focus function for automated autofocusing in microscopy.

1 INTRODUCTION

An automated microscope system is a powerful tool for biomedical research, especially in fluorescence microscopy (Boddeke, 1999). The control of focus is done through the use of motorized *z*-axis and an autofocus algorithm. The algorithm is an iterative procedure which steps along the *z*-axis to find the plane of the best focus. For each *z*-position an image is acquired from which a focus value is computed according to a certain focus function. Actually, the focus function is a function of the *z*-position. It says how sharp the image at the given *z*-position is. Ideally it has one clear maximum whose position corresponds to the *z*-position of the sharpest image.

According to (Groen et al., 1985) a useful focus function should fulfil several criteria such as: unimodality, accuracy, reproducibility, insensitivity to other parameters, the extremum be broadly tailed.

Although many focus algorithms have been proposed and compared (Groen et al., 1985; Firestone et al., 1991; Santos et al., 1997; Geusebroek et al., 2000; Sun et al., 2004; Bueno-Ibarra et al., 2005), the selection of an appropriate focus algorithm for specific conditions remains ad hoc and time consuming.

In fluorescence microscopy weak light signals are imaged in general and therefore more consideration is needed in choosing a focus function compared to bright field microscopy. Due to the placement of the objects (i. e., tissue or large cells) on the slides, the thickness of the objects, and possible bending of the slide, the objects normally do not lie in one plane and hence the focus function is usually not unimodal. This can consequently cause problems when searching for the position of the maximum focus, because many methods, such as Boddeke's algorithm (Boddeke et al., 1994), rely on unimodality of the focus function.

This paper is based on experiments with autofocusing simulated on a personal computer in the MATLAB environment. The results of these experiments are: assessment of current focus functions applied on special biomedical data; a proposed modification to existing autofocusing algorithm and its evaluation; and a method based on genetic programming designed for finding a suitable focus function especially for similar image data.

1.1 Materials and Methods

The data used were TOPRO-dyed high-resolution images of desmocytes. They were acquired in either confocal or standard (non-confocal) mode.

Cut outs 200×200 pixels[1] from 6 different samples were used for the experiments in order to reduce the time consumption. Each sample was acquired 99 times, once for each z-position. The size of the step between two successive images was manually chosen so that all the information needed for experiments would be captured. It was also dependent on whether the confocal mode was used or not. The z-positions are described by their natural order and treated as nondimensional values without respect to the step size used.

The information about the images is summarized in Table 1. In the last column there are the z-positions that are considered to be in focus. These numbers come from human perception and will be used as a reference.

Table 1: Information about image samples used for experiments.

No.	Step size (μm)	Mode	In-focus pos.
1	0.5	standard	53
2	0.5	confocal	33
3	0.2	confocal	43
4	0.2	confocal	42
5	0.5	confocal	47
6	1.0	standard	45

A technique called binning[2] (Netten, 1997; Kozubek et al., 1999; Boddeke et al., 1994) was applied on the samples. It is useful not only for reducing the spatial frequency, but it also minimizes the processing time[3]. The images after binning are presented in Figure 1.

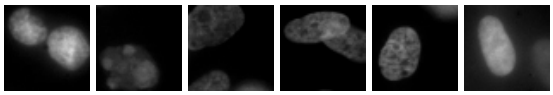


Figure 1: The in-focus images of testing samples 1 to 6 of TOPRO-dyed high-resolution images of desmocytes, after 4×4. binning.

2 STATE OF THE ART

In order to be able to implement an efficient and accurate autofocusing, a combination of a good focus function and a robust procedure for searching for the maximum is needed.

[1]This size is sufficient if suitable cut outs, i.e., those containing cells or other objects of interest, are selected.

[2]A group of adjacent pixes, here 4×4, is combined to form a superpixel.

[3]Now we have cut outs 50×50 pixels.

2.1 Focus Functions

16 classic focus functions from the literature were tested on the 6 samples (the list of these functions can be found in the appendix). The functions were implemented in MATLAB that is primarily optimized for computations with matrices, and therefore they were expressed using MATLAB-like functions with X being the acquired image. One of the main noticeable changes is the introduction of 2D convolution instead of pixel-wise operations that appear in the original notation of the functions. More details about the new notation will be given in section 4.

The error of a focus function is defined as a difference between the focus position given by this function and the reference focus position provided by human experts (see Table 1). By comparison of the errors of the focus functions under test, and after visual evaluation of the shape of these focus functions (with respect to the desired properties) it was concluded that five of them are applicable for such biomedical data. The two most suitable focus functions are the Variance (see Appendix for the formula) that was expressed in the form

```
div(sum(pow(minus(X,avg(X)),2)),area(X))
```

and the Vollath's F5 function (also in Appendix) expressed in the form

```
minus(sum(times(X,conv(X,V1))),
sum(times(X,conv(X,V2))))
```

where `V1=(1 0 0)` and `V2=(1 0 0 0 0)`.

Nevertheless, none of them is fully satisfactory. The mean error is never less than 4, the total error on the six samples is 24 or more. Detailed results can be found in Table 2.

Table 2: Errors of the 16 focus functions.

Func. No.	Samples 1	2	3	4	5	6	Total error	Mean error
1	15	3	2	0	1	11	32	5.3
2	12	2	22	1	2	25	64	10.6
3	12	2	2	0	2	6	24	4.0
4	44	3	0	1	2	23	73	12.2
5	46	3	2	1	1	25	78	13.0
6	12	2	2	1	2	23	43	7.2
7	26	3	0	0	2	17	48	8.0
8	15	2	2	0	2	7	28	4.6
9	49	1	21	41	46	33	191	31.8
10	15	2	2	0	2	7	28	4.6
11	49	1	0	41	46	33	170	28.6
12	49	66	18	3	45	33	247	41.2
13	49	17	18	1	46	34	165	27.5
14	12	2	2	0	2	6	24	4.0
15	15	2	2	0	2	7	28	4.6
16	24	48	41	57	48	53	271	45.2

2.2 Algorithm for Finding the Maximum Focus

For the purpose of automatic focusing we tested Boddeke's algorithm (Boddeke et al., 1994). It was originally designed for unimodal focus functions and works in three phases as follows:

The algorithm starts with the coarse phase at an arbitrary (but known) position z_0. First, two samples are acquired and their focus values are compared in order to determine the direction towards the focus. This may be a problem for focus functions with local minima, such as those applied on images with two or more natural in-focus planes. Then, more images are acquired until the last focus value is lower than the previous one. All the samples of the first phase are acquired evenly with the step Δz_{coarse}.

The starting position and the direction of the following fine phase, which is based on the same principle as the coarse phase, are given by the results of the coarse phase. The size step used during the fine phase is Δz_{fine}. Its size should correspond to the width of the so-called quadratic region (see (Boddeke et al., 1994) for details).

The last, refine phase of Boddeke's algorithm consists of capturing N_{refine} images around the position of the maximum focus function value found so far, fitting a parabola and finding its maximum.

The tuning of such an algorithm is a difficult goal: it concerns multiobjective optimization as we want to find the values of several parameters (Δz_{coarse}, Δz_{fine}, N_{refine}) while the number of images needs to be minimized and at the same time the accuracy and reliability to be maximized. Moreover, the goal is to make the algorithm robust to non-unimodal focus functions.

3 BODDEKE'S ALGORITHM IMPROVEMENT

Here we present the proposed modification of the algorithm described in 2.2:

1. In the coarse phase two images are acquired with a step Δz_{coarse} in order to find the direction towards the focus. Using the same step size, more images are acquired until the last focus value is lower than the previous one.
2. Interphase: the interval in which the focus lies is assured by acquiring two images, one to each direction, with a step Δz_{fine} from the z-position of the highest focus so far. The position of the higher focus function value of these two samples is the direction towards the focus. This step may help overcoming the bimodality that can occur.
3. In the fine phase images are acquired until the last focus value is lower than the previous one, with the step size Δz_{fine} from the position and to the direction determined by the previous phase. The two z-positions of the highest focus function values are passed to the refine phase.
4. The two z-positions from the fine phase together with three more (one in between and two on the sides of the interval, all being equidistant) are used for parabola fitting. The maximum computed from the parabola determines the position of the maximum focus.

The following parameters were determined based on empirical testing:

Table 3: Recommended parameters for the improved version of Boddeke's algorithm.

Parameter	Value
Δz_{coarse}	10
Δz_{fine}	6
N_{refine}	5

The total number of images needed depends strongly on the starting position z_0. Yet some estimation can be done, again based on some empirical tests: it varies from 8 to 14. Naturally, starting near to the in-focus position lowers the number of images needed. The efficiency of this algorithm is shown by the results of the algorithm applied together with the Variance focus function on the all 6 samples presented in Table 4. For each sample, all 99 starting z-positions were tested. The target is the position of the actual maximum of the focus function, irrespectively of the real in-focus position (we are assessing the algorithm itself now, not the accuracy of the Variance focus function, which is already known).

Table 4: Results of the improved version of Boddeke's algorithm using the Variance: $\Delta z_{fine.} = 6$, $\Delta z_{coarse} = 10$: a comparison of target, mean and standard deviation values.

Sample	1	2	3	4	5	6
Target	68	35	41	42	49	52
Mean	68.9	35.1	42.2	41.9	48.2	53.5
Sted. dev.	0.30	0.54	0.60	0.30	1.18	4.62

It can be concluded that the improved Boddeke's algorithm is a reliable procedure for adaptive searching the maximum of a function. If the focus function was more accurate, we would have been able to locate the position of the maximum focus quickly and precisely. Therefore, an effort should be made in order to

find better focus function since the focus function is currently the weakest link of the whole autofocusing.

4 FINDING THE FOCUS FUNCTION

The focus functions developed up to now are based on previous knowledge about the differences in information content in focused and unfocused image. But the space of all focus functions is potentially infinite. Nevertheless, we are looking for functions that are easy to compute. It means that these functions should be composed of relatively small number of operators and operands, such as the focus functions that were invented up to date. Still there may be plenty of other similar functions with unthought-of performance. Genetic programming (Koza, 1992) is a robust technique suitable for searching for optimal solution in a large state space. We will use it in an attempt to find a new focus function that will work well on noisy biomedical images.

Genetic programming works in the same way as genetic algorithms (Mitchell, 1996), but the population is composed of functions (or programs), that can be expressed in tree structure. The root represents the output of the function, the leaves represent the arguments. Inner nodes represent operators (here called elementary functions).

Genetic programming, identically to the theory of programming languages, faces the challenge of distinguishing between different data types. Strongly typed genetic programming (Montana, 1995) is a branch of genetic programming that finds inspiration in strongly typed programming languages, when dealing with this challenge. A modified version of this approach, adapted especially for our data types needs, will be presented here.

The MATLAB matrices of `doubles` are the only data types used for our computations. However, we need to work with matrices of any size, and distinguish between them. We will therefore define data types by pairs of integers that will correspond to the matrix sizes[4]. In addition, some functions have natural restrictions on the relations among the types (both output and input). We will therefore define them as *generic functions* (see (Montana, 1995)). Generic functions are functions that are able to work with more data types. They get instantiated during the tree building. This approach needs special attention to be paid during the process of tree generation (this will be described in detail in next sections).

[4]For example, a scalar will be of type $(1,1)$ etc.

Table 5: Table of functions, terminals and their types.

Function	Arity	Output	Input 1	Input2
plus	2	a,b	a,b	a,b
minus	2	a,b	a,b	a,b
times	2	a,b	a,b	a,b
mtimes	2	a,c	a,b	b,c
nmtimes	2	a,b	1,1	a,b
mntimes	2	a,b	a,b	1,1
divnum	2	a,b	a,b	1,1
divmat	2	a,b	a,b	a,b
pownum	2	a,b	a,b	1,1
conv	2	a,b	a,b	c,d
sum	1	1,1	a,b	–
abs	1	a,b	a,b	–
avg	1	1,1	a,b	–
log2	1	a,b	a,b	–
log10	1	a,b	a,b	–
uminus	1	a,b	a,b	–
area	1	1,1	a,b	–
min	1	1,1	1,a	–
max	1	1,1	1,a	–
hist	1	256,1	a,b	–
matrand	0	a,b	–	–
2	0	1,1	–	–
X	0	x,y	–	–

If the type (or one of the pair components) is generic, its value is represented by a *symbolic value* denoted by alphabetic letters *a*, *b*, etc. The *set of symbolic values* used for definition of a generic function represents the information we have at the moment about relations among the type components. During the process of creation of a new tree individual, there is a moment when an elementary function is selected for a node. If this elementary function is a generic function, we may only define the input and output types by symbolic values, but the important information about the relation is preserved. For example, if a binary function's output must be of the same type as its left-most input while the right-most input can be arbitrary, we define output in terms of symbolic values (a,b), the left-most input also by (a,b), but the right-most input is of type (c,d). The set of symbolic values in this case is $\{a,b,c,d\}$. During instantiation, each of these values is mapped to a specific numeric value (e.g., 3, 8, 5 and 1), but the property that the output type equals the right-most input's type holds. Of course, sets of symbolic values of different generic elementary functions are totally independent (e.g., *a* in the set of symbolic values of a function has nothing to do with *a* belonging to the set of symbolic values of any other elementary function within the same tree individual). Only when an elementary function

is assigned to a node that is direct descendant of another one, its instantiated output type and the relevant instantiated input type of the other function must fit. The aim is to have all the nodes fully instantiated after the tree is generated. The result of crossover, mutation etc. must be a valid function too.

The list of functions and terminals used together with their types specification is presented in Table 5. The terminal set consists of a variable X that represents the image input (its size (x,y) is known in advance), and a constant 2 that occurs very often in the classic focus functions. Function `matrand` of null parity generates random matrices of desired size in the program generation time. If the size is unspecified, `matrand` generates a matrix of random size between $(1,1)$ and $(10,10)$. Therefore its output type gets instantiated on a random basis. Some functions have predefined type values from the definition: for example, the function `sum` produces the sum of the elements of an arbitrary matrix and therefore its output type is fixed and its value is $(1,1)$. The function `hist` computes the histogram with 256 bins. The function `area` gives the multiplication of matrix sizes, `conv` is the 2D convolution of two matrices. The meaning of other functions is straightforward.

4.1 Generation of a New Tree

The most complicated part of the process lies in managing the data flow during the instantiation. The following definition will be necessary:

Definition 4.1 *A generic function is type-consistent if and only if it holds that if all the input types are fully instantiated, the output type is fully determined as well.*

An example of a function that does not fulfil this condition is a unary function of which input type is (a,b) and the output type is (c,d). The algorithm proposed can work only on type-consistent functions.

The tree is built recursively, in a depth-first manner. The data structures used by the algorithm are:

The table of functions and terminals – The table stores information about the generic functions and their types in terms of symbolic or predefined values, such as shown in Table 5. It is only read during the process.

The function instances in the nodes – This data structure is created at the same time when a node is created. The information gets updated gradually. First, the name of the assigned elementary function and its arity is stored here. Every time a type component of this function instance is instantiated, the value is stored here as well. Investigating the node, it can be assumed what types have already been instantiated.

The information flow concerning the types is done in four ways (Procedure A to D). Every time a new node is created, its output type might, but does not have to, be fully instantiated. (The root of any program that aims to become a focus function falls among those that have a specified output. It is the type $(1,1)$ because we want to measure a focus by a scalar value.) Based on this information, a function is randomly selected only from those that comply with the output type request. The number of the direct descendants of this node is now known.

The table of functions and terminals (Table 5) is read first in order to get the predefined values of type components, if there are any.

The components of the output type of the node are checked. If any of them is already instantiated, the symbolic values in the table are checked. If there is a symbolic value for this component of the output type, it means that we have instantiated it already. The specific value gets in the node to every type component that has the same symbolic value in the table (procedure B). The process can cause (partial) instantiation of both the input and output types of the node.

The information known at that moment is processed further when the left-most descendant is generated.

When the left-most subtree is completed, the process of elaborating this node continues. Because of the type consistence property the output type of the left descendant must be fully instantiated at that moment. If the left input type has not been instantiated before, it takes the value of the left-most descendant's output (Procedure C). This information may enrich the information about other types in the actual node, therefore procedure D is needed: it goes through the input types and if some of their values are newly instantiated, the symbolic values in the table are checked similarly to procedure B. The information goes to other components (including the output types) according to the symbolic value in the table.

The same process continues with all other descendants, including procedures C and D. At the latest when the last branch is finished, the output type of the actual node is fully instantiated, owing to the type consistent property and procedure D.

4.2 Genetic Programming Experiment Description

We used GPLAB (Silva and Almeida, 2003b), a genetic programmming toolbox for MATLAB, which

we modified in order to be able to handle strong typing. The initial population was generated using the ramped half-and-half method (Koza, 1992). Dynamic maximum tree depth, a technique for avoiding bloat (Silva and Almeida, 2003a), was also incorporated. The population size was 20, the dynamic limit and maximum limit were 5 and 20, respectively.

Standard genetic operators `reproduction`, `mutation` and `crossover` were used. The type constraint was resolved easily for mutation: a node was randomly selected, its type was found and a random tree was generated whose root's output type was the type needed. Crossover was performed only if the node selected from the first parent was present at least once in the second parent as well. In the case of multiple occurrence the final node for exchange was selected randomly.

The lexicographic parsimony pressure was used for selection together with elitism.

Fitness function was computed in the following way: first, candidate focus function was evaluated on the whole training set (an image sample acquired with various z-positions). Then a maximum was found, and the z-position that exhibits the maximum value was compared with the reference z-position. The difference is the error and therefore should be minimized. To eliminate functions that are constant around their maxima, the number of z-positions exhibiting the maximum value was added to the fitness function to disrate functions with undesirable shape.

More training sets are needed in order to prevent the genetic algorithm from guessing the correct z-position independently on the image itself. Due to high time consumption, the training set of three samples was used. It concerned samples 1, 3 and 6. The remaining three samples were used for testing.

4.3 Results

The resulting functions were then compared with one of the best classic functions, i. e. Variance, on a testing set of three different samples. Five new functions were giving comparable results. They even outperformed them on some samples. Their accuracy, i. e., how far a maximum was identified from the real one on the z axis, was evaluated together with their shapes and brevity.

One of the most interesting results is:

```
sum(plus(times(times(X,X),X),X))
```

which can be mathematically expressed as

$$f(z) = \sum_x \sum_y ((I(x,y,z))^3 + I(x,y,z))$$

The focus position given by that function is compared to the output of Variance in Table 6. In three cases (samples 3, 4 and 5) the genetic programming function was as successful as the two classic functions.

The graphs of the behaviour of this new focus function on all six samples can be found in Figures 2 and 3.

Table 6: Comparison of focus positions found by Variance and the genetic result.

Sample	Reference	Variance	Genetic result
1	53	68	71
2	33	35	36
3	43	41	41
4	42	42	42
5	47	49	49
6	45	52	55
Total error	–	28	35
Mean error	–	4.6	5.8

5 CONCLUSION

In this work several different tasks were accomplished. Their unifying topic is the autofocusing for automated microscopy that was simulated on a personal computer. First, 16 classic functions were tested on biomedical data. None of them was as accurate and reliable as would be necessary. Nevertheless, some of them are usable, mainly Variance and Vollath's F5. Second, Boddeke's algorithm was tested. It turned out that the algorithm can be modified so that it reliably finds the maximum of the function despite the potential bimodality. Some 10 acquired images are needed in average.

The weakest link is certainly the focus function itself. If this issue is solved, the automated focusing can become routine work without the need of human assistance. In order to find such a function, a preliminary genetic programming experiment was conducted. Its result is comparable to the Variance. This genetic programming application has shown the importance of solving the strong typing issue. An algorithm for tree generation using a variant of strongly typed genetic programming was designed, and an important property – the type-consistence – of elementary functions was recognized and defined.

5.1 Future Work

The genetic programming design could be improved by more careful selection of elementary functions, parameters etc. Running the program parallelly could enable using more individuals, generations and training samples. This altogether could result in a highly

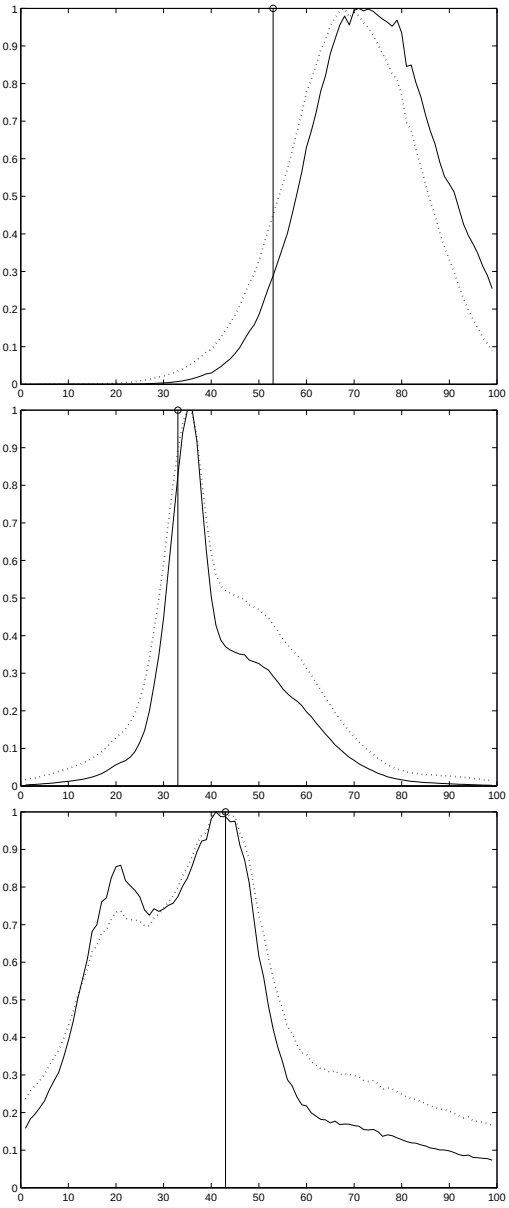


Figure 2: Genetic result (solid line) and Variance (dotted line) on testing samples 1 to 3 from top to bottom. The reference value is highlighted.

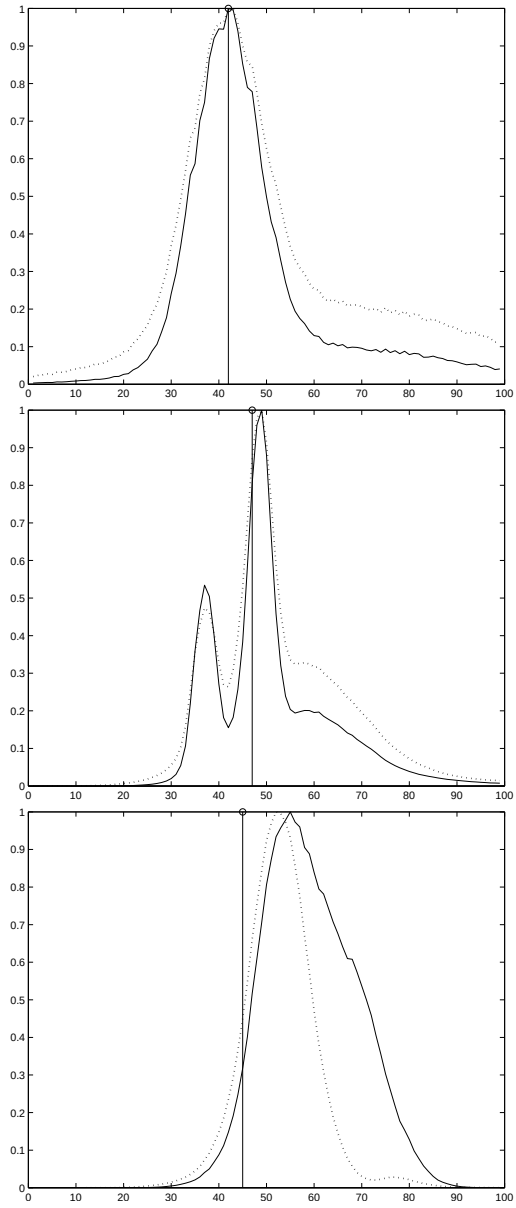


Figure 3: Genetic result (solid line) and Variance (dotted line) on testing samples 4 to 6 from top to bottom. The reference value is highlighted.

fit focus function of unprecedented performance that would discover the hidden features of images in focus. Finally, the results from PC simulations could be realized on a real automated microscope.

ACKNOWLEDGEMENTS

This work was supported by the Ministry of Education of the Czech Republic (Grants No. MSM-0021622419, No. LC535 and No. 2B06052).

REFERENCES

Boddeke, F. R. (1999). *Quantitative Fluorescence Microscopy*. PhD thesis, Technische Universiteit Delft.

Boddeke, F. R., van Vliet, L. J., Netten, H., and Young, I. T. (1994). Autofocusing in microscopy based on the otf and sampling. *Bioimaging*, 2:193–203.

Bueno-Ibarra, M. A., Álvarez Borrego, J., Acho, L., and Chávez-Sánchez, M. C. (2005). Fast autofocus algorithm for automated microscopes. *Optical Engineering*, 44.

Firestone, L., Cook, K., Culp, K., Talsania, N., and Preston, K. (1991). Comparison of autofocus methods for automated microscopy. *Cytometry*, 12:195–206.

Geusebroek, J., Cornelissen, F., Smeulders, A., and Geerts, H. (2000). Robust autofocusing in microscopy. *Cytometry*, 39:1–9.

Groen, F. C., Young, I. T., and Ligthart, G. (1985). A comparison of different focus functions for use in autofocus algorithms. *Cytometry*, 6:81–91.

Koza, J. R. (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press.

Kozubek, M., Kozubek, S., Lukášová, E., Marečková, A., Bártová, E., Skalníková, M., and Jergová, A. (1999). High-resolution cytometry of fish dots in interphase cell nuclei. *Cytometry*, 36:279–293.

Mitchell, M. (1996). *An Introduction to Genetic Algorithms*. Massachusetts Institute of Technology.

Montana, D. J. (1995). Strongly typed genetic programming. *Evolutionary Computation*, 3:199–230.

Netten, H. (1997). *Automated Image Analysis of FISH-Stained Cell Nuclei*. PhD thesis, Delft University of Technology.

Santos, A., Ortiz de Solórzano, C., Vaquero, J. J., Peña, J. M., Malpica, N., and del Pozo, F. (1997). Evaluation of autofocus functions in molecular cytogenetic analysis. *Journal of Microscopy*, 188:264–272.

Silva, S. and Almeida, J. (2003a). Dynamic maximum tree depth - a simple technique for avoiding bloat in tree-based gp. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2003)*.

Silva, S. and Almeida, J. (2003b). Gplab - a genetic programming toolbox for matlab. In *Proceedings of the Nordic MATLAB Conference (NMC-2003)*, pages 273–278.

Sun, Y., Duthaler, S., and Nelson, B. J. (2004). Autofocusing in computer microscopy: Selecting the optimal focus algorithm. *Microscopy Research and Technique*, 65:139–149.

APPENDIX

$I(x,y,z)$ ... The intensity of a pixel at a position (x,y) in an image I acquired at the z-position z

m,n ... The width and height of an image I

$\overline{I(z)}$ ... The average of the pixel intensity of an image I acquired at the z-position z, i. e., $\frac{\sum_x \sum_y I(x,y,z)}{m \cdot n}$

$h(i)$ ... Number of pixels of intensity equal to i in I

$p(i)$ ... The frequency of occurrence of pixels with intensity equal to i in an image I, i. e., $p(i) = \frac{h(i)}{mn}$

The list of the focus functions tested:

1. Absolute Gradient
   $f(z) = \sum_x \sum_y |I(x,y,z) - I(x,y-1,z)|$
2. Square gradient
   $f(z) = \sum_x \sum_y (I(x,y,z) - I(x,y-1,z))^2$
3. Netten's filter
   $f(z) = \sum_x \sum_y (I(x+1,y,z) - I(x-1,y,z))^2$
4. Energy Laplace
   $f(z) = \sum_x \sum_y C(x,y,z)^2$
   where $C(z) = I(z) * \begin{pmatrix} -1 & -4 & -1 \\ -4 & 20 & -4 \\ -1 & -4 & -1 \end{pmatrix}$
5. Laplacian
   $f(z) =$
   $\sum_x \sum_y (I(x,y-1,z) - 2I(x,y,z) + I(x,y+1,z))^2$
6. Tenegrad's function
   $f(z) = \sum_x \sum_y S_x(x,y,z)^2 + S_y(x,y,z)^2$ where
   $S_x(z) = I(z) * \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}$, $S_y(z) = I(z) * \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}$
7. Signal Power
   $f(z) = \sum_x \sum_y (I(x,y,z)^2$
8. Variance
   $f(z) = \frac{1}{mn} \sum_x \sum_y (I(x,y,z) - \overline{I(z)})^2$
9. Normalized Variance
   $f(z) = \frac{1}{mn(\overline{I(z)})^2} \sum_x \sum_y (I(x,y,z) - \overline{I(z)})^2$
10. Absolute Variance
    $f(z) = \frac{1}{mn} \sum_x \sum_y |(I(x,y,z) - \overline{I(z)}|)$
11. Normalized Absolute Variance
    $f(z) = \frac{1}{mn(\overline{I(z)})^2} \sum_x \sum_y |(I(x,y,z) - \overline{I(z)}|)$
12. Histogram Range
    $f(z) = \max_i(h(i) > 0) - \min_i(h(i) > 0)$
13. Histogram Entropy
    $f(z) = -\sum_i p_i \cdot \log_2 p_i$
14. Vollath's F_4
    $f(z) = \sum_x \sum_y I(x,y,z) \cdot I(x+1,y,z) -$
    $- \sum_x \sum_y I(x,y,z) \cdot I(x+2,y,z)$
15. Vollath's F_5
    $f(z) = \sum_x \sum_y I(x,y,z) \cdot I(x+1,y,z) - mn(\overline{I(z)})^2$
16. Spectral Analysis
    $f(z) = \sum_i p_i \cdot \log_{10} i$

DISTRIBUTED EMBEDDED SYSTEM FOR ULTRALIGHT AIRPLANE MONITORING

J. Kotzian and V. Srovnal, Jr.
Department of Measurement and Control, FEECS, VSB – Technical University of Ostrava, 17. listopadu 15
708 33 Ostrava – Poruba, Czech Republic
jiri.kotzian@vsb.cz, vilem.srovnal1@vsb.cz

Keywords: Embedded systems, graphic interface, industrial bus, industrial sensors, operating systems, control system design.

Abstract: This paper presents distributed embedded monitoring system that is developed for small aircrafts, sports and ultralights airplanes. System is made from modules connected by industrial bus CAN. This low cost system is trying to solve bad situation with many ultralights without any digital measurement unit due to their prices. The contribution shows basic architecture of the embedded monitoring system and presents some parts of hardware and software implementation. The interface between aviator and airplane is established using graphic user interface based on operating system uClinux.

1 INTRODUCTION

This paper is concentrated to avionic system especially to small sporting or ultralight airplanes. Here is basic information about small airplanes.

Figure 1: Ultralight airplane – illustrative photo.

Ultralight airplane is constructed for maximally 2 persons, with a stalling speed lower than 65km/h and a maximum flight weight of 450 kg. (Figure 1) Its price is much lower than professional airplanes but it is possible to fly for thousand kilometres. There are many standards describe ultralight league in many countries all over the world. The specification mentioned above is validated for Europe especially Czech Republic.

This project is focus on developing an alternative to high price products. The project has been developed together with the private company FALKON Electronics. We have developed the system architecture, which supports a flexible configuration. The configuration can grow from small, which measures basic values, to a wide range system. This is possible thanks to the module architecture.)

System is distributed into the independent modules that measure specific value on mechanical parts of the airplane. System is a configurable according type of airplane. The highest layer is graphic user module that represents received data on the LCD display. Sense of monitoring system is offer customers same facilities as have pilots in the professional aircrafts and make aviation more easier using low cost embedded electronic system.

2 AIRCRAFT MONITORING VALUES

There are two basic groups of values which can be measured. The first group includes flying values such as attitude and air speed, the second group is engine values as RPM and oil temperature. There is

also a third group for other values such as battery voltage, etc.

Fly values are the following:
- Attitude
- Altitude
- Airspeed
- Vertical Speed
- Gravitation
- GPS position
- Start, Fly, Actual Time

Engine values are the following:
- RPM
- Percent Power
- Oil temperature
- Oil pressure
- Cylinders temperatures
- Cylinders exhaust temperatures

The values that are mentioned above are only basic group for our purposes. Using embedded distributing system architecture we are able to extended whole system within other values.

3 MONITORING SYSTEM ARCHITECTURE

The real-time embedded control system is designed with a modular structure (Li and Yao 2003). This structure supports a flexible configuration. In terms of user requirements, the control system can be configured in different sizes and options. (Kotzian and Srovnal 2004) Several modules with different options were designed. All modules are connected to an industrial bus – so each module is the bus node. Except the GPS module, this is connected directly to the main control module.

The block diagram of a desk control and monitoring system with today's full configuration of prototype is shown in the Figure 2.

Designed modules are the following:
- Main control module
- User interface (LCD display) module
- Motor measuring values module
- Advanced avionic data module
- Black-Box module

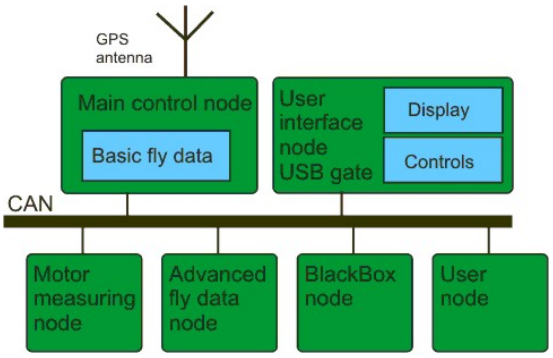


Figure 2: Block diagram of monitoring system

4 SYSTEM MODULES SPECIFICATION

The basic configuration contains only the User Interface Module and the Main Control Module. The Main Control Module has some basic inputs. Basic values are connected to these inputs, which have to be in the every airplane. The configuration can measure attitude, altitude, airspeed, gravitation, RPM, inside air temperature and battery voltage.

4.1 Communication Protocol

Monitoring modules are connected together by using an industrial bus (Sridhar 2003). This bus has to be highly reliable and have enough speed. Depending on these two main requirements a CAN bus was selected. The main reason is that the CAN has an extremely low probability of non-detected error. The versatility of the CAN system has proven itself useful in other applications, including industrial automation as well. A CAN bus is given the international standard ISO11898 which uses the first two layers of ISO/OSI model (CAN-CIA 2005). (Kotzian and Srovnal 2003)

4.2 Main Control Module

The main control node serves as master for all other nodes. (Arnold 2001) (Figure 3,4) Requesting values from other nodes are compared with given limits and stored in the local memory. The main module decides what information will be display and send to the user interface module by the CAN bus.

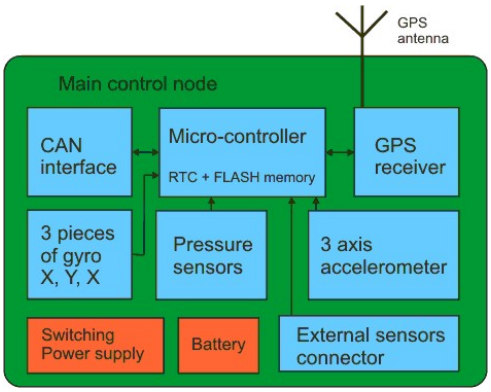


Figure 3: Main control node block diagram.

The main control module contains a real time clock and data flash memory for storing measured values and statistics. For measuring basic values the main module is equipped with the following measuring sensors:

- CSDX0811BARO for the altitude
- CSDX0025D4R for the air speed
- 3 x gyro sensor ADXRS401 for the attitude
- Accelerometer MMA7261Q

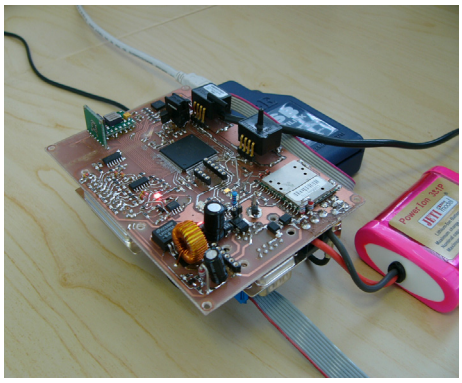


Figure 4: Main control module prototype (testing).

4.3 GPS Receiver

The GPS receiver is a small module with the passive antenna for receiving the position information from the global position system GPS. For its small size, good features and low price GPS Orcam 21SB was selected. The GPS module is integrated into the Main Control Module. External antenna is used due to the mounting possibility outside the plane (better GPS signal). The GPS module is connected by using a standard serial interface and standard GPS.

4.4 User Interface Module

The user graphic interface module serves as an interface between the user and all monitoring systems. There are two variants of the user interface module, an economical and comfort version.

The economical version includes the monochrome LCD Display GM62121 with the 320*248 pixels resolution. The economical version is equipped with a 16bit DSP controller without any operating system so it supports only necessary functions.

The comfort version includes a color TFT Display PD064 with a 640x480 pixels resolution. This version is equipped with a 32-bit processor (PPC or ColdFire) and operating system RT Linux (support MMU) (Hollabaugh 2002) or uClinux (MMU less). (Raghavan, Lad, Neelakandan 2006) There is also communication USB interface for storing the measured data in to the user USB devices (Service and diagnostic system). The operator panel of the comfort version is shown in the figure 5. The cheapest version uses a DSP controller as a display content computation and FPGA as a display driver.

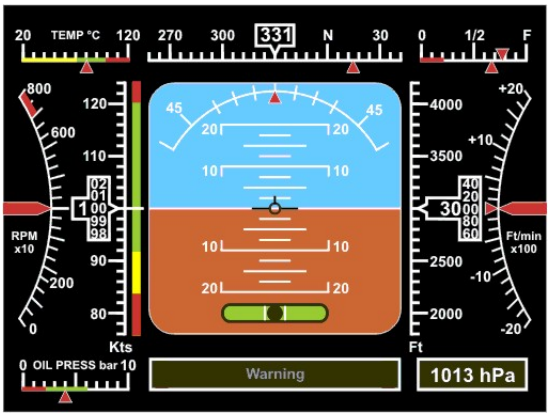


Figure 5: Main operator panel in cockpit.

It is possible to select the avionic screen, engine screen or GPS map screen.

The firmware is based on embedded operating system Linux using 240 MHz processor core. There are implemented 4 interfaces in the operating system. Two communication interface – CAN, RS232 and USB and one display interface FB (framebuffer) with the driver. Graphic system is built on GUI – microwindows or miniGUI.

Those graphic user interfaces using two graphic libraries SDL (Simple DirectMedia Layer) and OpenGL ES. SDL is cross-platform multimedia library designed to provide low level access to

audio, keyboard, mouse, joystick, 3D hardware via OpenGL, and 2D video framebuffer. OpenGL ES is a royalty-free, cross-platform API for full-function 2D and 3D graphics on embedded systems. Software implementation is base on Eclipse Workbench.

4.5 Black-Box Module

The black-box module controls all traffic on the CAN bus. It reads data from CAN messages and stores data in the local memory. The black-box module is equipped with its own RTC timer and stores time together with the CAN data. There is no other connection to this module with such high reliability.

5 DEVELOPING SYSTEM CONFIGURATION

The project is presently in last of developing state. We are beginning with the final versions of the modules. The Main Control Module is designed in its final version and is under testing.

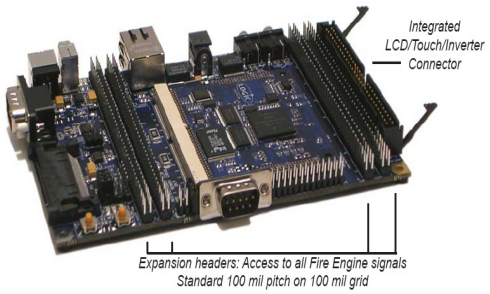


Figure 6: Evaluation kit for operator panel in cockpit.

Low cost version is based on EvbDSP module with a DSP56F805 controller, six 8-bits ports, CAN, SPI and serial interfaces and FPGA graphic driver interface (User Interface Module). The EvbHCS12 module is equipped with a MC9S12DP256 controller (Main Control Module). This module supports a wide range of interfaces: CAN, LIN, serial, SPI, I2C and six 8-bit ports. The smallest module is the EvbHCS08 with a MC9S08GT60 controller, LIN, SPI and serial interfaces, temperature and humidity sensors and four 8-bit ports (I/O devices). The comfort version has different User Interface Module that is based on 32-bit ColdFire controller (EvbMCF5329) with an integrated display controller – GUI (Figure 6). For EvbMCF5329 was used Linux BSP.(Yaghmour 2003)

6 CONCLUSION

The development and realization of the avionic control and monitoring system for ultra-light airplanes is very important for increasing the safety and security of pilots. The number of accidents of ultra-light planes is too high, especially during the starting and lending stages of flights. A low cost flight control and monitoring system is the best way to cut down on the number of accidents. The developed monitoring system can be configured from a minimal version to a wide system based on customer requirements.

ACKNOWLEDGEMENTS

This project is being completed at VSB - Technical University of Ostrava, Czech Republic. The work and the contribution were supported by a project of the Grant Agency of Czech Republic – 102/65/0571 Architectures of embedded system networks and department of measurement and control.

REFERENCES

Arnold K. (2001) Embedded Controller Hardware Design. *LLH Technology Publishing USA 2001, ISBN 1-878707-52-3*

Sridhar T. (2003) Design Embedded Communications Software. *CMP Books, San Francisco 2003, ISBN 1-57820-125-X*

Raghavan P., Lad A., Neelakandan S. (2006) Embedded Linux System Design and Development. *Auerbach Publication USA 2006, ISBN 0-8493-4058-6*

Li Q., Yao C. (2003). Real-Time Concepts for Embedded Systems, *CMP Books, San Francisco 2003, ISBN 1-57820-124-1*

Hollabaugh, Craig., (2002). Embedded Linux, *Pearson Education, Indianapolis 2002, ISBN 0-672-32226-9*

Yaghmour, Karim., (2003). Building embedded Linux systems, *O'Really & Associtates, Sebastopol 2003, ISBN 0-596-00222-X*

Kotzian J. and Srovnal V. (2004). Development of Embedded Control System for Mobile Objects Using UML. *In : Programmable Devices and Systems 2004-IFAC Workshop, Krakow, IFAC WS 2004 0008 PL, ISBN 83-908409-8-7, p.293-298*

Kotzian J. and Srovnal V. (2003). Can Based Distributed Control System Modelling Using UML. *In: Proceeding International Conference IEEE ICIT 2003, Maribor, Slovenia, ISBN 0-7803-7853-9, p.1012-1017*

HEURISTIC ALGORITHMS FOR SCHEDULING IN A MULTIPROCESSOR TWO-STAGE FLOWSHOP WITH 0-1 RESOURCE REQUIREMENTS

Ewa Figielska
Warsaw School of Computer Science, Lewartowskiego 17, 00-169 Warsaw, Poland
efigielska@poczta.wwsi.edu.pl

Keywords: Heuristics, Linear Programming, Flowshop scheduling, Parallel unrelated machines, Resource constraints.

Abstract: This paper deals with the problem of preemptive scheduling in a two-stage flowshop with parallel unrelated machines at the first stage and a single machine at the second stage. At the first stage, jobs use some additional resources which are available in limited quantities at any time. The resource requirements are of 0-1 type. The objective is the minimization of makespan. The problem is NP-hard. Heuristic algorithms are proposed which, while solving to optimality the resource constrained scheduling problem at the first stage of the flowshop, select for simultaneous processing jobs according to rules promising a good (short) schedule in the flowshop. Several rules of job selection are considered. The performance of the proposed heuristic algorithms is analyzed by comparing their solutions with the lower bound on the optimal makespan. The results of computational experiments show that these heuristics are able to produce near-optimal solutions in short computation time.

1 INTRODUCTION

During the last years, the flowshops with multiple processors (FSMP) also called hybrid flowshops, have received considerable attention from researchers (e.g. Gupta 1988; Chen 1995; Haouari and M'Hallah, 1997; Brah and Loo (1999), Linn and Zhang, 1999; Ruiz and Maroto, 2006).

In this paper, we extend multiprocessor flowshop scheduling research by including resource constraints. We consider the problem of scheduling in a two-stage flowshop where jobs use additional renewable resources, which are available in limited quantities at any time. This problem can be described as follows. There are n preemptive jobs to be processed at two stages in the same technological order, first at stage 1 then at stage 2. At stage 1 there are m parallel unrelated machines, stage 2 has one machine. A job upon finishing its processing at stage 1 is ready to be processed at stage 2; it may be processed at stage 2 when the machine is available there, or it may reside in a buffer space of unlimited capacity following stage 1 until the machine at stage 2 becomes available. At stage 1, a job can be processed on any of the parallel machines, and its processing times may be different on different machines. The processing times of job j ($j=1,\ldots,n$) are equal to p_{ij} (if it is executed on machine i ($i=1,\ldots,m$)) and s_j time units, respectively, at stage 1 and at stage 2. The processing of a job on a machine of stage 1 may be interrupted at any moment and resumed later on the same or another machine. A job during its processing at stage 1 does not need a resource or uses one unit of this resource (0-1 resource requirements). There are l types of resources. A resource of type r ($r=1,\ldots,l$) is available in an amount limited to W_r units at a time. The total usage of resource r at any moment by jobs simultaneously executed on parallel machines cannot exceed the availability of this resource. The objective is to find a feasible schedule which minimizes makespan, $C_{\max}$, which is equal to the maximum job completion time at stage 2.

The considered problem is NP-hard in the strong sense since the problem of preemptive scheduling in the two-stage flowshop with two identical parallel machines at one stage and one machine at another is NP-hard in the strong sense (Hoogeveen et al., 1996).

The heuristic algorithms proposed for the considered problem, while solving to optimality the resource constrained scheduling problem at the first stage of the flowshop, select for simultaneous processing jobs according to rules promising a good (short) schedule in the flowshop. Several rules of job selection are considered.

The problem under consideration arises in real-life systems that are encountered in a variety of industries, e.g. in chemical, food, cosmetics and textile industries. These systems are often subjected to some additional resource constraints for example on the availability of the additional resources such as skilled labour and tools. Preemption of jobs usually results in shortening the schedule. The problem with parallel unrelated machines at the first stage and a single machine at the second stage may arise in a manufacturing environment in which products are initially processed on any of parallel machines and then each product must go through a final testing operation, which is to be carried out on a common testing machine.

2 FRAMEWORK OF THE HEURISTIC ALGORITHMS

The proposed heuristic algorithms proceed in the following steps:

1. A linear programming (LP) problem is solved to minimize time T needed for finishing all jobs at stage 1 of the flowshop under relaxed resource constraints over time T. As a result, the minimal value of T and the values of the time t_{ij} ($i = 1,\ldots,m,\ j = 1,\ldots,n$) during which job j is processed on machine i are obtained.
2. Using the values of t_{ij} obtained in Step 1 as well as the values of s_j (s_j is the processing time of job j at stage 2), weights, w_j, for all jobs are determined on the basis of 6 different expressions presented in Table 1.
3. The schedule at stage 1 of the flowshop is constructed in the form of a sequence of partial schedules using the values of T, t_{ij}, and w_j. In a partial schedule at most m (m is the number of machines) jobs are assigned to machines for simultaneous processing during some period of time so that resource constraints are satisfied at every moment. The consecutive partial schedules are created in subsequent iterations of an iterative procedure. Assignment of jobs to machines in a partial schedule is found maximizing the weighted assignment $\sum_{i=1}^{m}\sum_{j=1}^{n} w_j v_{ij}$ (v_{ij} = 1 if job j is processed on machine i in a current partial schedule, and 0 otherwise) under resource constraints. In each created partial schedule, conditions on optimality formulated in (Slowinski, 1980, 1981) are satisfied.
4. Completion times of jobs at stage 1 are calculated.
5. A schedule on the machine of stage 2 is constructed using the values of s_j, and ready times of jobs at stage 2, which are equal to corresponding completion times at stage 1.

Table 1: Weights used in the heuristic algorithms.

Algorithm	Weight w_j of job j
A1	1
A2	random number from $U[0,1)$
A3	$\frac{\min_{k=1,\ldots,n}\{Z_k\}}{Z_j}$
A4	$\frac{s_j}{\max_{k=1,\ldots,n}\{s_k\}}$
A5	$\frac{s_j}{Z_j}\ \frac{\min_{k=1,\ldots,n}\{Z_k\}}{\max_{k=1,\ldots,n}\{s_k\}}$
A6	$\frac{\min_{k=1,\ldots,n}\{Z_k\}}{Z_j}+1$ if $Z_j \le s_j$, and $\frac{s_j}{\max_{k=1,\ldots,n}\{s_k\}}$ if $Z_j > s_j$

Z_j is the processing time of job j at stage 1, $Z_j = \sum_{i=1}^{m} t_{ij}$. In A3, A4 and A5 the maximal value of w_j is equal to 1, in A6 the maximal w_j is equal to 2 if $Z_j \le s_j$, and 1 if $Z_j > s_j$.

3 ILLUSTRATIVE EXAMPLE

To illustrate the problem and the solution method we present the following example. Consider the case of the two-stage flowshop with 2 machines at stage 1 and a single machine at stage 2. The number of jobs n=10, the resource availability at any moment, W_1=1. Job processing times and resource requirements are shown in Figure 1.

job processing times at stage 1:

job	machine 1	2
1	6	10
2	14	9
3	12	8
4	13	10
5	6	12
6	23	16
7	6	8
8	13	9
9	19	22
10	8	23

job processing times at stage 2:

job	machine 1
1	1
2	8
3	10
4	6
5	6
6	7
7	2
8	9
9	1
10	1

resource requirements at stage 1:

job	
1	1
2	1
3	0
4	1
5	0
6	0
7	0
8	0
9	1
10	1

resource availability = 1

Figure 1: Data for an illustrative example.

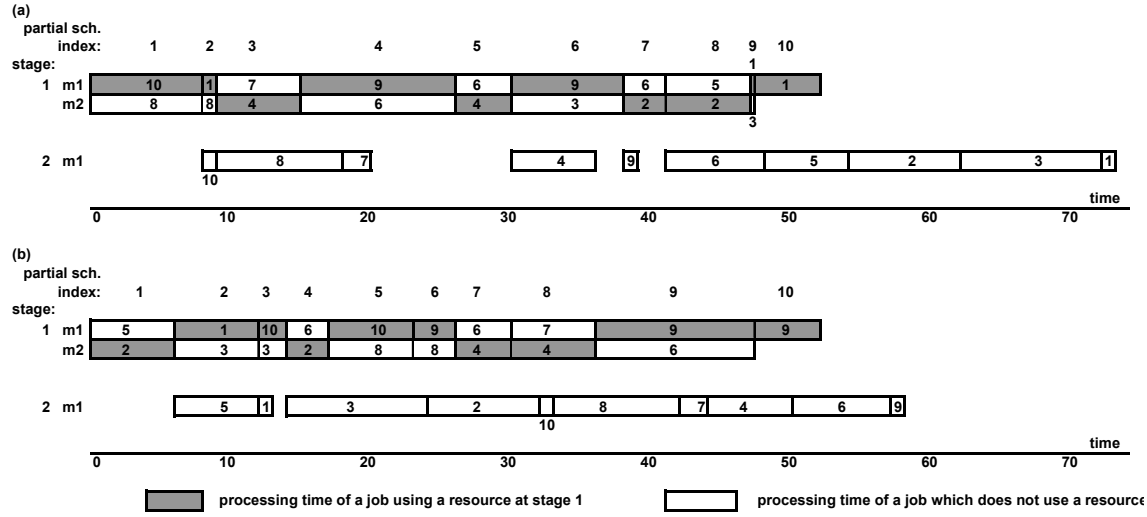


Figure 2: An illustrative example. The resulting schedules: (a) a schedule obtained by heuristic algorithm A1, (b) a schedule obtained by heuristic algorithm A5.

Figure 2 presents two flowshop schedules for this instance, one created by algorithm A1 (Figure 2a) and another created by A5 (Figure 2b). The first stage schedules in Figures 2a and 2b are composed of 10 partial schedules. In each of these partial schedules at most 2 jobs are processed simultaneously and the total resource usage does not exceed the resource availability, $W_1 = 1$, e.g. in the partial schedule of index 1 in Figure 2a, jobs 10 and 8 are processed simultaneously and use at every moment 1 and 0 units of the resource, respectively. The first stage schedules in Figures 2a and 2b have the same length, but completion times of jobs in these schedules are different. For example, in the first stage schedule in Figure 2a, job 10 finishes its processing at stage 1 at 8 time units and in Figure 2b - at 23 time units. This results in different lengths of the flowshop schedules provided by A1 and A5.

4 COMPUTATIONAL STUDY

In this section, the results of a computational experiment conducted to evaluate the performance of the proposed heuristic algorithm are presented. 720 randomly generated instances were created and examined. Instances were generated for $n = 50$, 100, 150, and 200, $m = 2$, 4, and 6, and for one resource type. The resource availability W_1 was set at $m/2$, and for 55% of jobs resource requirements were set at 1. Processing times at stage 2, s_j, were generated from $U[1,100]$ ($U[a,b]$ denotes the discrete uniform distribution in the range of $[a,b]$) for all instances. Processing times at stage 1, p_{ij}, were generated from 9 ranges: $U[30,120]$, $U[45,180]$, and $U[60,240]$ for instances with $m = 2$, $U[90,360]$, $U[120,480]$, and $U[150,600]$ for $m = 4$, $U[150,600]$, $U[200,800]$, and $U[250,1000]$ for $m = 6$. The ranges

for processing times were selected so as to include the cases when the length of the optimal schedule at stage 1, C_1^*, is close to the sum of job processing times at stage 2, $\sum_{j=1}^{n} s_j$.

As an effectiveness measure we use the relative percentage deviation of a heuristic solution from the lower bound on the optimal makespan defined as

$$\delta = \frac{C_{\max} - LB}{LB} \times 100\%$$

where $LB = \max\{LB_1, LB_2\}$, where $LB_1 = C_1^* + \min_{j=1,\ldots,n}(s_j)$ and $LB_2 = \min_{i=1,\ldots,m,\, j=1,\ldots,n}(p_{ij}) + \sum_{j=1}^{n} s_j$, C_1^* is the minimal makespan at stage 1.

Table 2: Computational results.

		δ (%)					
n	m	A1	A2	A3	A4	A5	A6
50	2	1.38	1.40	0.06	0.75	0.06	0.04
		2.78	3.35	1.30	0.54	0.12	0.06
		4.15	4.43	2.74	0.27	0.20	0.14
	4	3.26	1.94	0.31	0.46	0.15	0.40
		7.21	4.60	3.77	0.60	0.52	0.60
		7.68	6.25	5.81	0.82	0.89	0.74
	6	2.65	1.39	0.25	0.52	0.16	0.52
		8.24	5.16	4.64	0.87	0.77	0.84
		8.90	7.80	6.87	1.55	1.68	1.51
100	2	0.74	0.51	0.03	0.39	0.02	0.01
		1.80	2.49	0.60	0.21	0.04	0.02
		1.52	1.33	1.00	0.03	0.05	0.02
	4	1.26	1.08	0.08	0.26	0.08	0.25
		5.06	3.18	1.87	0.18	0.22	0.18
		3.45	2.70	2.50	0.17	0.21	0.16
	6	2.03	1.01	0.06	0.23	0.07	0.25
		5.64	4.03	2.46	0.40	0.31	0.42
		4.02	4.03	3.76	0.47	0.47	0.43
150	2	0.21	0.42	0.02	0.19	0.02	0.00
		1.62	1.16	0.28	0.19	0.02	0.01
		0.95	0.90	0.91	0.02	0.02	0.02
	4	0.71	0.61	0.00	0.22	0.01	0.16
		4.24	2.05	1.44	0.15	0.10	0.19
		1.94	1.60	1.37	0.06	0.07	0.06
	6	0.76	0.49	0.04	0.20	0.01	0.20
		4.14	2.36	1.86	0.24	0.17	0.24
		2.66	2.46	2.17	0.18	0.20	0.19
200	2	0.46	0.42	0.01	0.21	0.02	0.00
		1.10	0.99	0.17	0.30	0.01	0.01
		0.69	0.48	0.38	0.01	0.01	0.01
	4	0.60	0.44	0.04	0.19	0.03	0.10
		2.15	1.70	0.86	0.12	0.05	0.12
		1.47	1.22	1.16	0.05	0.05	0.05
	6	0.83	0.25	0.02	0.10	0.03	0.10
		2.46	1.57	0.90	0.12	0.07	0.12
		1.75	1.67	1.64	0.10	0.10	0.10

The results of a computational experiment are presented in Table 2 All entries in this table are average values over 20 instances.

From Table 2, we can observe that deviations, δ, significantly decrease, as the number of jobs grows, and they increase with the number of machines. We can see that algorithms A3, A4, A5, and A6 always outperform A1 and A2, and A4, A5, and A6 produce near-optimal solutions. On the average over the entire collection of instances, relative deviations of the heuristic makespan from its lower bound are equal to 2.79%, 2.15%, 1.43%, 0.32%, 0.19%, and 0.23% for A1, A2, A3, A4, A5, and A6, respectively.

The CPU times are small for all the heuristic algorithms and equal to about 0.3, 1.5, 3, 4.5, and 6.5 seconds for n=50, 100, 150, and 200, respectively.

REFERENCES

Brah, S.A. and L.L. Loo (1999). Heuristics for scheduling in a flow shop with multiple processors. *Europ. J. of Opernl Res.*113,113-112

Chen, B. (1995). Analysis of classes of heuristics for scheduling a two-stage flow shop with parallel machines at one stage, *J. of Opernl Res. Soc.* 46, 234-244.

Gupta J.N.D. (1988). Two stage hybrid flowshop scheduling problem. *J. of Opernl Res. Soc.* 39, 359-364.

Haouari, M. and R. M'Hallah (1997). Heuristic algorithms for the two-stage hybrid flowshop problem, *Oper. Res. Let.* 21, 43-53.

Hoogeveen J.A., J.K. Lenstra and B. Veltman (1996). Preemptive scheduling in a two-stage multiprocessor flow shop is NP-hard. *Europ. J. of Opernl Res.* 89, 172-175.

Linn, R, W. Zhang (1999). Hybrid flow shop scheduling: a survey. *Comp. and Ind. Engin.* 37, 57-61.

Ruiz, R., C. Maroto (2006). A genetic algorithm for hybrid flowshops with sequence dependent setup times and machine eligibility. *Europ. J. of Opernl Res.* 169, 781-800.

Slowinski, R. (1980), Two approaches to problems of resource allocation among project acticities – A comparative study. *J. Opl Res. Soc.* 31, 711-723.

Slowinski, R. (1981), L'ordonanacement des taches preemptives sur les processeurs independants en presence de ressources supplementaires. *RAIRO Inform./Comp. Science*, vol.15, No.2, 155-166.

A GROWING FUNCTIONAL MODULE DESIGNED TO TRIGGER CAUSAL INFERENCE

Jérôme Leboeuf Pasquier
Departamento de Ingeniería de Proyectos, CUCEI, Universidad de Guadalajara
Apdo. Postal 307, CP 45101, Zapopan, Jalisco, México
jleboeuf@dip.cucei.udg.mx

Keywords: Artificial brain, epigenetic robotics, autonomous control, learning system, growing neural network.

Abstract: "Growing Functional Modules" constitutes a prospective paradigm founded on the epigenetic approach whose proposal consists in designing a distributed architecture, based on interconnected modules, that allows the automatic generation of an autonomous and adaptive controller (artificial brain). The present paper introduces a new module designed to trigger causal inference; its functionality is discussed and its behavior is illustrated applying the module to solve the problem of a dynamic maze.

1 INTRODUCTION

1.1 The Epigenetic Approach

According to Epigenesis, introduced in Developmental Psychology by Piaget (Piaget, 1970), the emergence of intelligence in a system requires such system to have a physical presence (embodiment) that allows an interaction with the environment (situatedness), furthermore this system should hold an "epigenetic developmental process" in charge of developing some specific skills to fulfill particular goals. If extrapolating this theory to Computer Science, Robotics constitutes the proper application field as it provides both embodiment and situatedness; then, an intelligent system performing as the robot's controller could emulate the "epigenetic developmental process".

Formally introduced in (Leboeuf, 2005), Growing Functional Modules (GFM) constitutes a prospective paradigm founded on this epigenetic approach. As a result, a GFM controller gradually acquires the specialized abilities while trying to satisfy some induced internal goals, which can be interpreted as motivations.

1.2 The Concept of GF Module

In input, a module receives requests; each request corresponds to the directive of reaching a specific state. The corresponding finite set of states is initially empty, but it gradually increases integrating as a new state, any distinct values provided by feedback. Furthermore, each module is assigned a set of commands that allows it acting on its environment, either directly by positioning some actuators or indirectly by sending requests to other modules. States transitions are achieved triggering these commands. Hence, each module enclosed a dynamic structure, typically a network of cells that gradually grows to memorize the correlations between these state transitions and a corresponding sequence of commands.

The engine of the module is in charge of retrieving an optimal sequence of transitions connecting the current state to the requested one and then, replicating it while triggering the corresponding sequence of commands (propagation). Obviously, the environment, commonly the real world, does not present a deterministic behavior due mainly, to an incomplete perception (the finite set of sensors reflects only a fraction of the reality); but also to errors associated to sensing (like round off, precision of the sensors, mechanical imperfections and external disturbances). So, when feedback exhibits some minor differences between the predicted behaviour and the obtained one, an adaptation mechanism is in charge of adjusting the current transitions; while, in case of major differences, a new sequence of transitions may be computed and then replicated.

GF modules have no previous learning phase; learning and adaptation may occur at any time when

propagation does not fit the previously acquired behaviour. Nevertheless, to avoid a limitless growing of the internal structure, this structure must converge when interacting with a stable environment, i.e. an environment in which for this specific module the same causes produce the same effects during a certain period of time.

As a result, a module constitutes an autonomous entity able by its own to perform a specific class of control. For instance, two recent papers describe the RTR-Module and its improved version (Leboeuf, 2006) designed to perform basic automatic control; its behavior is illustrated by the application to an inverted pendulum.

1.3 The Concept of GFM Architecture

In addition to the ports dedicated to input requests, to feedback and to output commands previously mentioned, all modules incorporate an input-output inhibition port that allows a module to prevent propagation in another one. As a result, the interconnection of modules through these communication ports allows the elaboration of a multi-purposes architecture.

An illustration of a GFM architecture including six modules of three different types is described in (Leboeuf, 2005). The corresponding controller shows its ability to discover and handle the potential functionalities of a virtual, mushroom shaped robot including the control of the single leg's steps, the direction of the body and the orientation of the hat. To induce learning of such functionalities, at least one global goal is required at the top of the architecture. In the case of the mushroom shaped robot, such induced motivation corresponds to light seeking.

In accordance with this approach, the GFM "programming process" consists of graphically designing the GFM architecture, i.e. interconnecting a set of functional modules. Afterward, two C++ source files are automatically generated; they contain the constructor of the controller and a description of the serial communication protocol between the controller and its associated application. Compiling these two files and linking them with the GFM library produce the GFM controller that initiates its activity when connected to the corresponding application (virtual or real). As a result, this paradigm, though still in a development phase, brings forward the possibility of setting up an autonomous and adaptive controller replacing the traditional programming task by the design and training of a distributed architecture.

As a prospective computer science paradigm, GFM offers several relevant aspects:

- First, memory is the exclusive product of a learning process;
- Second, the acquired knowledge and its processing engine are indivisible;
- Third, adaptation to changing environments is intrinsic;
- Fourth, all the learning process is guided by the satisfaction of global goals (that may be interpreted as motivations).

Therefore, as a system's architecture, GFM has a propensity to introduce a more natural concept of memory and knowledge processing.

2 THE CI-MODULE

2.1 Concept

Earlier, the conception and development of a new GF module arises to satisfy the necessity of controlling some specific hardware. Presently, the proposal of its creation comes out from the importance of causal inference in cognitive psychology. This assertion and the subsequent statements are inspired by the theories of cognitive psychology concerning learning and memory exposed in (Anderson, 1999).

There, causal inference appears to play a fundamental role concerning adaptation since an organism able to discover the cause of some phenomenon also acquires the ability of predicting and/or producing some behavior in its environment and consequently, it is able to control some particular aspects in order to satisfy its needs. Moreover, causal inference allows facing more complex situations than, for example, associative learning and therefore seems to be involved in many cognitive levels from simple action-reward activity to language acquisition.

Such important role of causal inference justifies its introduction as a new module of the Acting Area in charge of triggering the corresponding sequence of actions that satisfies a specific request from a higher module (a process referred in the previous paragraph as "producing some behavior"). Its counterpart belonging to the Sensing Area and in charge of interpreting multiple feedbacks from sensors (referred as "predicting some behavior") is still under study and will not be described in the present paper. Hence, active and passive functionalities of causal inference mentioned above

are partitioned to be integrated to the GFM architecture.

In robotics, GFM's field of application, causal inference should play a key role in planning because it requires the robot to elaborate an optimum path from an initial state corresponding to its current situation toward a final one that matches the requested conditions.

2.2 Achieving Propagation

The internal structure of a CI-Module is represented as a state graph, as illustrated on figure 1. The transition from one state to another is obtained by triggering the associated command. For example, from the current state, identified with black color and labeled with number '7', it is possible to reach the state labeled '6' by triggering the command c_4, at least when the universe is fully deterministic. Then, different sequences of commands allow, starting from the current state, reaching the goal state labeled with number '9' and identified with a double circle. For example, the sequences (c_1 c_1 c_2), (c_1 c_1 c_1 c_4), (c_2 c_4 c_1 c_2), (c_3 c_2 c_4) among many others, comply with this purpose. Internally, all states are defined and validated as a result of the feedback values and the goal state is given by the input request. Besides, all commands belong to the set assigned to the module during the designing phase.

As a consequence, propagation consists of finding and applying the optimal sequence of commands to reach the goal. The qualifier "optimal" refers to the sequence offering the lowest cost while considering that each command has an associated cost. This problem is analogous to the search of the shortest path in a graph considering that the weights given here, represent the cost associated with the commands. With the condition, presently verified, that all weights are positive, Dijkstra's algorithm (Dijkstra, 1959) always encounters the optimal solution to this problem. The principle of this algorithm consists in repeatedly adding the most economical edge from the currently visited nodes to any unvisited ones, and iterating this process until reaching the goal node. Several authors have proposed alternative solutions with better performances but with distinct hypothesis (Cooper et al., 2000); so, at present, propagation is still guided by an implementation of the Dijkstra's algorithm. In case, all commands have the same cost, then the best path would have the lowest number of transitions; in case several paths have the same number of transitions (see blue and green paths represented on figure 2) then the first one provided by the shortest-paths algorithm will be chosen.

The costs mentioned previously in the propagation process are obtained as follows: each time it triggers a command, any GF acting module provided an extra feedback value that either refers the effort produced by the actuator to realize the corresponding action or, in case of a lower module, integrates all the subsequent efforts leading to comply the corresponding request. This is an important aspect as any module must prefer the lowest solution. This feedback value produced by a lower level set up the cost associated to the transition. This cost may vary in time due to external effects (see section 2.4), thus it must be updates permanently as follows: each time a transition is applied successfully, a new cost is assigned as the half of the sum of the current cost and the returned value.

2.3 Learning Transitions

To comply with the GFM paradigm, a module must build up its internal structure from nothing, only using the guidance of the feedback. At the

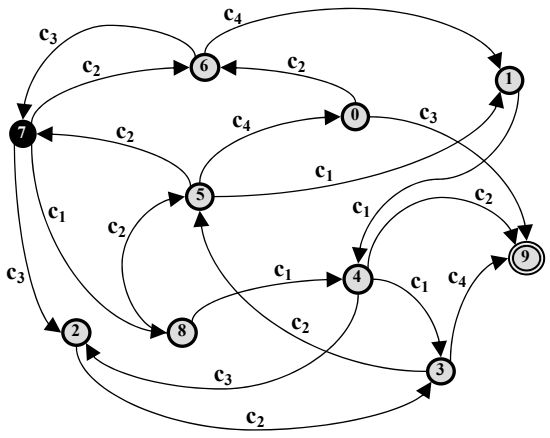


Figure 1: Graph of the internal structure of a CI-Module.

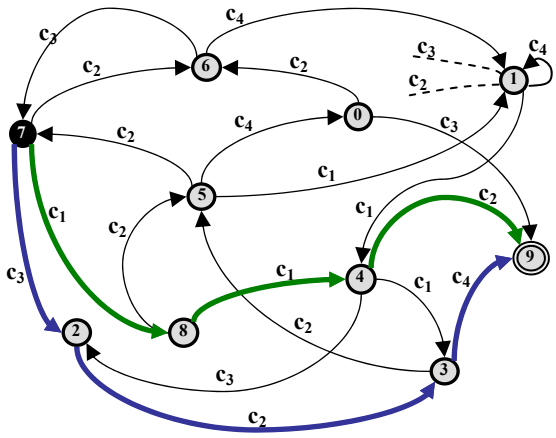


Figure 2: Two optimal paths to the goal state.

beginning, the initial state, given by the feedback, corresponds to the local perception of the initial value of a particular sensor. Then, from this state, a randomly selected command is triggered and eventually a new state discovered; this process is re-iterated each time there is no path leading to the requested goal, including the particular case when this requested goal is still unknown and sometimes non-existent; furthermore, each different value given by the feedback is interpreted as a new state.

The graph resulting from this process is stored in memory and permanently updated. According to the illustration of figure 2, the only transition from state '1' is correlated to the command c_1, it comes out that the commands c_2 and c_3 have not been tested yet and that the command c_4 do not produce any transition. Commands like c_4 should be ignored as they do not generate a state transition nevertheless they must be occasionally triggered to corroborate this fact. If required, the graph is updated because, as exposed next section, the environment is rarely deterministic.

2.4 Dealing with Non Determinism

The previous description of the CI-Module considers the interaction with a deterministic environment, nevertheless in the real world, an action often produces uncertain effects due to

- The imprecision of the sensory system that engenders an incomplete representation of the environment;
- The presence of external and hidden effects;
- The eventual presence of other entities that possibly alter the current representation.

2.4.1 First Mechanism

Therefore, each time a command is triggered, the new current state is determined by the supplied feedback. If this state corresponds to the predicted one, then propagation may continue in accordance with the computed path; on the opposite, a new path, starting from the new current state, determined by the feedback value. This mechanism is fundamental to deal with non determinism; an illustration is presented figure 3 where the predicted path starting from state '7', passing by '8' and '4' to reach the goal state '9' must be recomputed because state '5' is specified by feedback instead of the predicted state '4'; next, a new path is computed that consists of states '5', '0' and finally '9'.

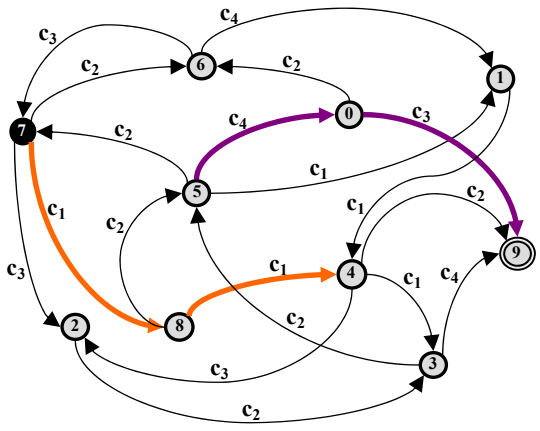


Figure 3: Re-computing the predicted path.

2.4.2 Second Mechanism

Moreover, in accordance with the previously mentioned uncertainty of the environment, our representation should not be deterministic but must offer a distribution of probabilities corresponding to the main outcomes of a command triggered from a specific state. Such distribution of probabilities should reflect the previous experience.

Several paradigms, in particular those based on probability calculus like Markov Chains (Wai-Ki and Michael, 2006) or Bayesian Networks (Jansen, 2001), build faithful representation of the environment. Nevertheless, with regard to our problem, they present the major inconvenient to constitute passive processes, in the sense that they do not act directly on the environment to establish a resultant representation. In fact, these paradigms are fine candidates to the sensing counterpart of the CI-Module mentioned in section 2.1. Presently, an alternative process has been implemented and this is for two reasons: first to increase efficiency since one criterion for growing functional modules is to be a low cpu and memory consumer; next, because only short term memory is required since it is assumed that the response of the environment may frequently change and so, its resulting feedback. In other

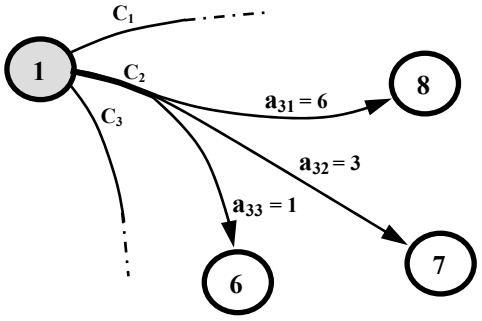


Figure 4: Memorizing different transitions generated by a specific command c_2.

words, the second implementation will favor fast adaptation over long time memory.

The implementation of this process consists in memorizing a small finite number t of different transitions generated by a specific command c from a given state s during a few last p intents and in assigning the number a_i of achievements to each transition. For example, the transition depicted on figure 4, reflects that during the last p attempts of triggering the command c_2 from the state *'1'*, six transitions reached the state *'8'*, three reached the state *'7'* and one the state *'6'*. Consequently, state *'8'* that actually presents the highest reliability is considered as the expected transition and is used to compute the predicted path to reach the goal state. It must be noticed that the sum of the achievements is not equal to the number of attempts p because while the successful alternative is incremented, all others are decremented at the same time. Further, when an alternative reaches the value p then, this means that no other alternatives had appeared during the last p attempts. Therefore, p reflects the sensibility to cope with the changes that presents the environment: a lower value of p produces a more dynamic behavior.

The transitions' reliability is updated in the following way: each time a command is triggered, the resulting transition is compared to the t memorized transitions. If found, its corresponding reliability is incremented while the reliabilities of all the other alternatives are decremented. If not found, this new transition replaces the lowest memorized one, and its reliability is set to 1. This mechanism obviously favors the most recent outcomes of a transition contrarily to a model based on historical probability. In some way, experiments practiced on rats dealing with mazes support this choice: the last successful outcome obtains a relative predominance over historical experience.

2.4.3 Additional Mechanisms

Three additional mechanisms are described in this section; they all have been introduced to improve the propagation in the internal structure of the CI-Module.

The third mechanism consists of triggering, from time to time, a command that, until now, has not produced any transition from a particular state; like for example in figure 2, the command c_4 triggered from state *'1'*. This mechanism allows the detection of potential paths that, on the contrary, would remain ignored after being the object of an initial failure. The implementation consists in periodically triggering unproductive actions if any exist. In practice, when a state has been activated a predefined number of times, an unproductive action is randomly selected and triggered.

The fourth mechanism consists of allowing computation of the path using the best and the second best transitions when their reliabilities are relatively close. This mechanism reflects the fact that two close reliabilities denote two good potential transitions. But, this mechanism presents the inconvenience to consequently increase the processing time necessary to compute the optimum path; therefore, its application is only optional.

The fifth mechanism has been introduced to take into account the reliability of a transition. When applying the Dijkstra algorithm, the choice of the optimum path is achieved taking into account the cost of the transition as previously described, but also considering its reliability. This mechanism clearly indicates that a longer but more reliable path should be preferred to a shorter but more erratic one. The implementation of this mechanism consists of linearly increasing the cost according to the reliability of the transition: the cost is doubled when the reliability is minimal (i.e. equal to 1) and unchanged when the reliability is maximum (i.e. equal to p). Moreover, introducing reliability in the evaluation of the potential paths contributes to make the previous one obsolete.

A simplified pseudo-algorithm of the final process is given figure 5; the presence of the different mechanisms is also indicated.

3 COMPLIANCE

Any Growing Functional Module must satisfy three main requirements to comply with the GFM paradigm and thus, to be interconnected with other modules and integrated into a GFM architecture. These requirements, discussed in the present section, contemplate the growing of the internal structure, the interconnection with other modules and the contribution of the module.

First, the internal structure of any module must be initially nonexistent and designed to gradually grow as the result of some learning mechanisms. Moreover, these mechanisms must allow the permanent adaptation of the internal structure although this structure must stabilize when performing in a stable environment. The implementation of the CI-Module describes in section 2 satisfies this requirement: when the initial request is received as input, the first state is originated; then new structures are created only in

response to new or inconsistent feedback which means that, if the environment is stable, the internal structure compound of states and transitions will not grow. This leads to a conclusion about the convergence of the structure since the local adaptation of the transitions has no growing effect in term of memory.

Second, the parameters that give a description of the application to the module are reduced to the standard ones including: the set of output commands, the feedback value associated to the current state and an input request with no specified range. Due to the object oriented implementation, any module derives from a common class that includes all input-output ports; in consequence, the CI-Module automatically fits the interconnection requirement.

Thirdly, to illustrate its functionality and contribution to the GFM architecture, the module must achieve some generic control task, operating as an autonomous entity that is able to communicate with the application through the standard protocol. The satisfaction of this requirement is exhibited next section where the CI-Module is employed to achieve the learning of a dynamic maze.

In addition to the previous requirements, it is imperative to consider memory and processing costs associated with the implementation of a module since a GFM architecture is supposed to integrate many modules. Concerning the CI-Module, the main processing cost involves the algorithm that computes the shortest path. The current implementation, based on the standard definition of the Dijkstra's algorithm, offers a regular performance. Nonetheless, (Fredman & Tarjan, 1987) demonstrates that using a new structure called F-Heaps the problem may be solved in near-linear time: $O(n.log(n) + m)$ where m and n respectively represents the number of edges and vertexes. Furthermore, (Matias et al., 1994) introduces the notion of "approximate data structure" and proposes a faster solution of the shortest path problem with the condition of tolerating a small amount of error. This offers a convenient alternative in case of elevated cpu requirements as, in the case of GFM modules, precision and exactitude requirements are not essential because errors are considered part of the learning process. Besides, the memory cost is proportional to the number of states including for each state a maximum number of p transitions, p being the number of authorized commands; so, to store a state and its p transitions, only $2p+28$ bytes are required with p commonly less than 10. Thus, both, memory and cpu costs, appear to be proportional to the number of achievable states. Such number is in turn, related to a sufficient learning time to produce a reliable behavior that could be defined as a fixed minimum percentage of correct responses of the system, typically ninety per

```
WHILE ContinueControl
  TotalCost←0;
  WaitFor(Request);
  Get(Feedback);
  FindOrCreate(CurrState,Feedback);
  Found←Search(ShortPath); M2,4,5
  IF Found
    Compute(Trans,PredState,Command); M3
  ELSE
    Choose(Command);
  WHILE CurrState#Request AND Command
    Trigger(Command);
    Get(Feedback,Cost);
    TotalCost←TotalCost+Cost;
    FindOrCreate(CurrState,FeedBack);
      IF CurrState#PredState M1
        IF CurrState
      Update(Trans,CurrState,Cost); M2,5
          Search(ShortPath); M2,4,5
        ELSE
          Create(Trans,CurrState,Cost);
          Choose(Command);
      ELSE
        Reinforce(Trans,Cost);
  Compute(Trans,PredState,Command); M3
Return(TotalCost);
```

Figure 5: Simplified pseudo-algorithm of the internal structure process corresponding to the CI-Module where the mentioned mechanisms X are referred as M_X.

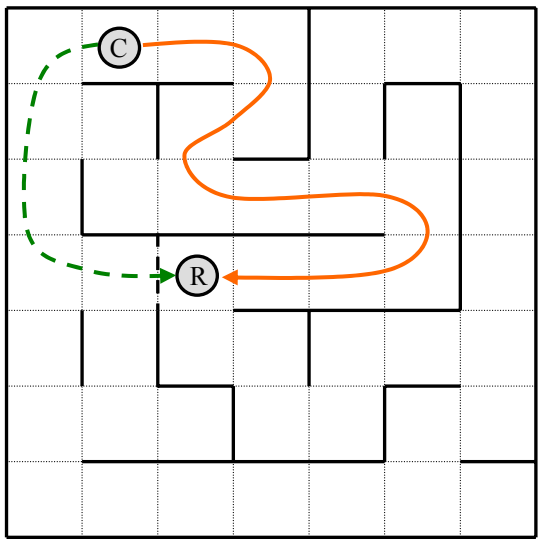


Figure 6: A typical maze used as an application where letters 'C' and 'R' stands respectively for the current and the requested positions.

cent. Hence, the recommendation should be given to the programmer, i.e. designer of the architecture, to estimate the number of discrete states that will generate a particular feedback from the environment.

Note that it could be assumed that a higher number of commands will produce a larger internal structure though there is no strict mathematical relation between these concepts. This last assertion may be inferred from the following example: all the possible commands on actuators may be integrated within a tiny structure of only two states representing "mobility" and "immobility". Nevertheless, it could be assumed that, when integrating the CI-Module with an architecture, it is convenient to associate a reduced set of commands.

In conclusion, due to its compliance, the corresponding C++ implementation of the CI-Module may be added to the GFM library that integrates all the fundamental components and permits the design of an architecture for a GFM controller.

4 APPLICATION

As mentioned before, each time a GF module is developed a corresponding application must be programmed and tested in parallel because the module is supposed to be able to perform as a controller on its own. Such an application represents the challenge that directs the development of a specific module.

In cognitive psychology, mazes constitute natural tools to investigate causal inference. The obtained results in this area have inspired in some way the present study as, for example, the reference introduced at the end of section 2.4.2. Actually, a maze is the application defined to evaluate the performance of the CI-Module. Nevertheless, the behaviour of a rat in a maze is more complex than a causal inference based control system because others cognitive abilities like spatial representation take part in the process.

In the current application, maze's positions correspond to states whose current one is provided as feedback to the controller; meanwhile the set of actions is composed of the elementary moves in each of the four possible directions: north, south, east and west. The task of the module consists of learning the configuration of a particular maze as illustrated on figure 6 and learning to satisfy a request that stipulates which position to reach. Thus a classic maze is implemented with three additional characteristics:

- First, the internal walls constituting the maze may shift randomly from time to time;
- Second, some "sliding effect" sometimes alters the response to an action, in other words the maze is not fully deterministic.
- Third, the cost of each move is a function of its direction.

Experimentally, the CI-Module is perfectly able to perform the task of learning and solving the maze. This is obvious since the maze offers a mirror representation of the state graph that describes the internal structure of the module. In fact, initially, the maze has been conceived as an illustration of the module's expectations and then, it has been employed to guide its development. Hence, for the designer, the maze serves as a reference of the ability of the CI-Module.

The major difficulty has been to deal with the walls' shifts in absence of tactile or visual perception and thus, to discover a potential shorter path as the emergent one indicated with dashed lines on figure 6. In absence of any complementary perception, the solution, given in section 2.4.3 consists of occasionally, triggering apparently inefficient commands, thus giving the system the ability of discovering new opportunities. Despite its slowness, this mechanism satisfies the initial requirement of adaptability inherent to the architecture.

The tests have been realized running the single module on a computer and the maze application on another, both connected by a serial connection implemented with the GFM standard protocol. The results show that the version of the CI-Module described in this paper, comply with all its expected functionalities.

Finally, the importance of feedback timing must be considered. In practice, the perception of a causal inference depends of the proximity in space and mostly time between an action and an effect. So, as the delay increases, the consistency of the relation decreases. The feedback is provided to the CI-Module once the command has been fully performed. In the case of commands triggered by modules that involve requests to lower modules, the feedback is provided when all the subsequent commands have been applied and, consequently this will increase the feedback's delay. However, the reaction of the environment is not always instantaneous; for example, "the production of heat as resulting from switching on a lamp" represents a complex inference as a result of the extended delay that occurs between both events. In such case, the sensing Area is supposed to help providing the

interpretation of the phenomenon, but, at this time there is no evidence to support this hypothesis.

5 CONCLUSIONS

The present paper introduced a novel Growing Functional Module designed to perform causal inference. Its conception and implementation are fully described and its behavior illustrated by the application to a dynamic maze problem.

The foremost conclusion is that this module shows to be fully functional and compliant with the requirements of any Growing Functional Module. Consequently, it has been added to the GFM library that incorporates all the components employed to automatically build GFM controllers. Besides, the internal structure of the CI-Module exhibits the intrinsic qualities of the epigenetic approach: both processes of learning and propagation are guided by a (restricted) feedback of the environment.

Forthcoming works derived from the present one include a module conceived as a counterpart of the CI-Module designed to integrate the Acting Area. The internal structure of this future module of the Sensing Area is based on a Finite State Automaton. Furthermore, an improved version of the present module is also under study; it has a more complicated internal structure based on an Interpreted Petri Net.

REFERENCES

Piaget, J., 1970. *Genetic Epistemology.* Series of Lectures, Columbia University Press, Columbia, New York.

Wai-Ki, C. & Michael, K.N., 2006. *Markov Chains: Models, Algorithms and Applications*. Springer Eds.

Jansen, F.V., 2001. *Bayesian Networks and Decision Graphs*, Springer Eds.

Leboeuf, J., 2005. *Growing Functional Modules, a Prospective Paradigm for Epigenetic Artificial Intelligence.* Lecture Notes in Computer Science 3563. Springer, pp. 465-471.

Leboeuf, J., 2006. *A Growing Functional Module Designed to Perform Basic Real Time Automatic Control*. Publication pending in Lecture Notes in Computer Science, Springer.

Leboeuf, J., 2006. *Improving the RTR Growing Functional Module*. Proceedings of the 32nd Annual Conference of the IEEE Industrial Electronics Society, IECON'06 Paris, pp. 3945-3950.

Leboeuf, J., 2005. *Applying the GFM Prospective Paradigm to the Autonomous and Adaptive Control of a Virtual Robot*. Lecture Notes in Artificial Intelligence, Springer 3789, pp. 959-969.

Dijkstra, E.W., 1959. *A Note on Two Problems in Connexion with Graphs*. Numerische Mathematik 1, pp. 269-271.

Fredman, M.L. & Tarjan, R.E., 1987. *Fibonacci Heaps and their Use in Improved Network Optimization*. Journal of the ACM Vol. 34, pp 596-615.

Matias, Y., Vitter J.S. &Young, N.E., 1994. *Approximate Data Structure with Applications.* Proceedings of the ACM-SIAM Symposium, pp. 187-194

Cooper, C., Frieze, A., Mehlhorn, K. & Priebe, V., 2000, *Average-Case Complexity of Shortest Path Problems in the Vertex-Potential Model*.

Anderson, J.R., 1999. *Learning and Memory, an Integrated Approach*. Wiley Eds.

SPECIAL SESSION ON FROM PLANNING TO CONTROL OF MANUFACTURING SYSTEMS

CHAIRS:
JEAN-LOUIS BOIMOND
JEAN JACQUES LOISEAU

SUPERVISORY CONTROL OF HEAP MODELS USING SYNCHRONOUS COMPOSITION

Jan Komenda
Mathematical Institute, Czech Academy of Sciences, Brno Branch Zizkova 22, 616 62 Brno, Czech Republic
komenda@ipm.cz

Jean-Louis Boimond and Sébasten Lahaye
LISA Angers, 62, Avenue Notre Dame du Lac, 49000 Angers, France
{boimond,lahaye}@istia.univ-angers.fr

Keywords: Heap models, synchronous product, supervisory control, dioid algebra.

Abstract: Heaps models are powerful models for concurrent timed discrete event systems. They admit linear description using dioid algebras. Inspired by supervisory control of logical discrete event systems we introduce parallel composition of heap models, called synchronous product, to formally describe the action of supervisor (represented by another heap model) on the system. This additional explicit concurrency for naturally concurrent heap models is useful for studying supervisory control in the algebraic framework of dioid algebras. Timing aspects of supervisory control, *i.e.* optimal timing of the controller, is studied based on residuation theory.

1 INTRODUCTION

There are two major streams in control theory of Discrete Event (dynamical) Systems (DES). The first stream, known as supervisory control theory, has been introduced by Wonham and Ramadge for logical automata (e.g. (Ramadge and Wonham, 1989)). The second stream more particularly deals with the class of timed Petri nets, called timed event graphs, based on linear representation in the (max,+) algebra. Being inspired by papers on (max,+) automata (e.g. (Gaubert and Mairesse, 1999), (Gaubert, 1995)), which generalize both logical automata and (max,+)-linear systems, it is interesting to develop a control method for (max,+) automata by considering supervisory control approach. However the time semantics of the parallel composition operation (called supervised product) we have proposed for control of (max,+) automata in (Komenda et al., 2007) are different from the standard time semantics for timed automata or timed Petri nets. One has to increase the number of clocks in order to define a synchronous product of (max,+) automata viewed as 1-clock timed automata. This goes in general beyond the class of (max,+) automata and makes powerful algebraic results for (max,+) automata difficult to use.

The results of (Gaubert and Mairesse, 1999) suggest however an alternative for the subclass of (max,+) automata corresponding to safe timed Petri nets, where synchronous product is standard composition of subnets through shared (synchronization) transitions. The intermediate formalism of heap models enables a letter driven (max,+)-linear representation of 1-safe timed Petri nets. Therefore it is interesting to work with heaps of pieces instead of (max,+)-automata and introduce a synchronous composition of heap models that yields essentially reduced nondeterministic (max,+)- automata representation of synchronous composition of corresponding (max,+)-automata. This way we obtain representations allowing for use of powerful dioid algebras techniques and the reduced dimension of concurrent systems at the same time: the dimension of synchronous product of two heap models is the sum of each models dimensions, while the dimension of supervised product of (max,+)-automata is the product of the individual dimensions, which causes an exponential blow up of the number of states in the number of components.

The extension of supervisory control to timed DES represented by timed automata is mostly based on abstraction methods (*e.g.* region construction turning a timed automaton into a logical one). On the other hand abstraction methods are not suitable for (max,+) or heap automata, because their timed semantics (when weights of transitions are interpreted as their minimal durations) are based on the earliest

possible behavior similarly as for timed Petri nets. The method we propose avoids any abstraction and works with timed DES (TDES) represented by heap models. Similarly as for logical DES our approach to supervisory control is based on the parallel composition (synchronous product) of the system with the supervisor (another heap model).

Our research is motivated by applying supervisory control on heap models, which are appropriate to model 1-safe timed Petri nets. This is realized by using the synchronous product: the controlled system is the synchronous product of the system with its controller (another heap model). The algebraization of the synchronous product that is translated into idempotent sum of suitable block matrices together with a linear representation of composed heap models using its decomposed morphism matrix is then applied to the control problem for heap models.

The paper is organized as follows. Algebraic preliminaries needed in the paper are recalled in the next section. In Section 3 are introduced heap models together with their synchronous product and their modelling by fixed-point equations in the dioid of formal power series. Section 4 is devoted to the study of properties of synchronous product of heap models that will be applied in Section 5 to supervisory control of heap models. Residuation theory can be used in supervisory control of heap models.

2 DIOID ALGEBRAS

Basic algebraic structures and their properties that will be used in the paper are briefly presented in this section.

An idempotent semiring is a set M endowed with two inner operations. A commutative and associative addition denoted $\oplus$ that has a unit element ε and satisfies the idempotency condition ($\forall a \in M:\ a \oplus a = a$). A second operation, called multiplication, and denoted $\otimes$ is associative and has a unit element e, distributes over $\oplus$ both on the left and on the right, and $\forall a \in M:\ a \otimes \varepsilon = \varepsilon \otimes a = \varepsilon$. An idempotent semiring is said to be commutative if the multiplication $\otimes$ is commutative.

There is a naturally defined partial order $\preceq$ on any idempotent semigroup, namely, $a \preceq b$ if and only if $a \oplus b = b$. An idempotent semiring M is called to be complete if any nonempty subset A of M admitss a least upper bound denoted by $\bigoplus_{x \in A} x$ and the distributivity axiom extends to infinite sums. Idempotent semirings are usually called dioids.

Let $\mathbb{N}$ denote the set of natural numbers with zero. In complete dioids the star operation can be introduced by the formula

$$a^* = \bigoplus_{n \in \mathbb{N}} a^n$$

with $a^0 = e$. Matrix dioids are introduced in the same manner as in the conventional linear algebra.

The simplest examples of commutative dioids are number dioids such as $\mathbb{R}_{max} = (\mathbb{R} \cup \{-\infty\}, max, +)$ with maximum playing the role of idempotent addition, denoted by $\oplus$: $a \oplus b = max(a,b)$, and conventional addition playing the role of multiplication, denoted by $a \otimes b$ or ab when no mistake is possible. Examples of non commutative dioids are matrix number dioids, formal languages and formal power series.

Let us recall from (Baccelli et al., 1992) and (Gaubert, 1992) the following results.

Theorem 2.1 *Let D be a complete dioid, x, a, b in D and*

$$x = x \otimes a \oplus b. \tag{1}$$

The least solution to equation (1) exists and is given by $b \otimes a^*$.

Lemma 2.2 *Let D be a complete dioid, a, b in D. Then $(a \oplus b)^* = (a^* b)^* a^* = a^* (b a^*)^* = b^* (a b^*)^* = (b^* a)^* b^*$.*

In the sequel we will work with the dioid of formal power series in the noncommutative variables from A and coefficients from $\mathbb{R}_{max}$ (corresponding to time). The standard notation A^* is used for the free monoid of finite sequences (words) from A. The empty word is denoted by 1. Formal power series form a dioid denoted $\mathbb{R}_{max}(A)$, where addition and (Cauchy or convolution) multiplication are defined as follows. For two formal power series in $\mathbb{R}_{max}(A)$:
$s = \oplus_{w \in A^*} s(w) w$ and $s' = \oplus_{w \in A^*} s'(w) w$, we have

$$s \oplus s' = \oplus_{w \in A^*} (s(w) \oplus s'(w)) w \text{ and}$$

$$s \otimes s' = \oplus_{w \in A^*} (\oplus_{uv = w} s(u) s'(v)) w.$$

This dioid is isomorphic to the dioid of generalized dater functions from A^* to $\mathbb{R}_{max}$ *via* a natural isomorphism similarly as the dioid $\mathbb{Z}_{max}(\gamma)$ of formal power series, used to study Timed Event Graphs (TEG), is isomorphic to the dioid of daters from $\mathbb{Z}$ to $\mathbb{Z}_{max}$. This isomorphism associates to any $y:\ A^* \to \mathbb{R}_{max}$ the formal power series $\oplus_{w \in A^*} y(w) w$ in $\mathbb{R}_{max}(A)$. The zero and identity series are denoted by ε and e, respectively, because it will be clear from the context whether a number dioid or dioid of formal power series is meant. Let us recall that $\forall w \in A^*:\ \varepsilon(w) = -\infty$ and

$$e(w) = \begin{cases} 0 & \text{if } w = 1 \\ -\infty & \text{if } w \neq 1. \end{cases}$$

We consider in the sequel the complete version of $\mathbb{R}_{max}(A)$ with coefficients in $R_{max} \cup \infty$. The notion of

projection will be needed. Similarly as for formal languages, (natural) projection will be introduced such that it will be morphism with respect to both $\oplus$ and $\otimes$.

Finally, residuation of matrix multiplication will be needed in Section 5. Residuation theory generalizes the concept of inversion for mappings that do not necessarily admit an inversion, in particular those among ordered sets. If $f : C \to D$ is a mapping between two dioids, in most cases there does not exist a solution to the equation $f(x) = b$. Instead of solutions to this equation, the greatest solution to the inequality $f(x) \preceq b$ or the least solution to the inequality $b \preceq f(x)$ are considered. In the case these exist for all $b \in D$, the mapping f is called residuated, and dually residuated, respectively. The corresponding mappings that to any $b \in D$ associate the greatest, resp. least, solutions of the corresponding inequalities are called residuated, resp. dually residuated mappings. In this paper only residuated mappings of matrix multiplication are used. The residuated mapping of the left matrix multiplication, *i.e.* the greatest solution to the inequality $A \otimes X \preceq B$, is denoted $A\backslash B$. Similarly, the residuated mapping of the right matrix multiplication, *i.e.* the greatest solution to the inequality $X \otimes A \preceq B$, is denoted B/A. Recall from (Gaubert, 1992) that for matrices $A \in D^{m\times n}, B \in D^{m\times p}, C \in D^{n\times p}$ over a complete dioid D

$$A\backslash B \in D^{n\times p} : \ (A\backslash B)_{ij} = \bigwedge_{l=1}^{m} A_{li}\backslash B_{lj}$$

$$B/C \in D^{m\times n} : \ (B/C)_{ij} = \bigwedge_{k=1}^{p} B_{ik}/C_{jk}.$$

We recall the notation $\bigwedge_{s\in S} s$ for the infimal element of a set $S \subseteq D$ (recall that $\bigoplus_{s\in S} s$ is the corresponding supremal element of S when lattice structure of a complete dioid is considered).

3 SYNCHRONOUS COMPOSITION OF HEAP MODELS

Let us first recall the definition of time extension of heap models (Gaubert and Mairesse, 1999), also called task-resource systems, which model an important class of TDES. Formally, a heap model is the structure $\mathcal{H} = (A, R, r, l, u)$, where

- A is a finite set of pieces (also called tasks).
- R is a finite set of slots (also called resources).
- $r : A \to Pwr(R)$ gives the subset of slots required by a piece. It is assumed that $\forall a \in A : \ R(a) \neq \emptyset$.
- $l : A \times R \to \mathbb{R} \cup \{-\infty\}$ is function such that $l(a,r)$ gives the height of the lower contour of piece a at the slot r.
- $u : A \times R \to \mathbb{R} \cup \{-\infty\}$ is function such that $u(a,r)$ gives the height of the upper contour of piece a at the slot r. By convention, $u \geq l$, $l(a,r) = u(a,r) = -\infty$ if $r \notin R(a)$, and $\min_{r\in R(a)} l(a,r) = 0$.

The dynamics of heap models is described by row vectors $x(w)_r, \ w \in A^*, \ r \in R$ corresponding to the height of the heap $w = a_1 \ldots a_n$ on slot $r \in R$. It has been shown in (Gaubert and Mairesse, 1999) that the upper contour of a heap w, denoted by $x(w)$, and the overall height of the heap w, denoted $y(w)$, are given by the following letter driven dynamic equations:

$$x(1) = (0 \ldots 0) \quad (2)$$

$$x(wa) = x(w)\mu(a) \quad (3)$$

$$y(w) = x(w)(0 \ldots 0)^T, \quad (4)$$

where

$$\mu(a)_{sr} = \begin{cases} 0, & \text{if } s = r \text{ and } s \notin R(a) \\ u(a,r) - l(a,s), & \text{if } r \in R(a) \text{ and } s \in R(a) \\ -\infty, & \text{otherwise} \end{cases} \quad (5)$$

is the morphism matrix associated to the heap model $\mathcal{H}$. It is shown in (Gaubert and Mairesse, 1999) that heap models are special (max,+)-automata with input and output functions as row, resp. column vectors of identity elements, and the morphism matrix above. The (max,+)-automaton given by the triple input function, output function, and the morphism matrix described above is then called heap automaton. Therefore we can associate heap models with special (max,+)-automata called heap automata.

We assume in the definition of synchronous product below that there are no shared resources between two heap models. Otherwise stated: resources are shared only by tasks within individual heap models. This requirement is best understood if one considers safe timed Petri nets (which can be viewed (Gaubert and Mairesse, 1999) as particular heap models), where synchronous compositions of subnets is realized by synchronizing shared transitions (in heap models tasks), while the set of places (in heap models resources) of the individual subnets are disjoint.

Definition 3.1 *(Synchronous product) Let* $\mathcal{H}_i = (A_i, R_i, r_i, l_i, u_i)$, $i = 1,2$ *be two heap models with* $R_1 \cap R_2 = \emptyset$. *Their* synchronous product *is the heap model*

$$\mathcal{H}_1 \| \mathcal{H}_2 = (A_1 \cup A_2, R_1 \cup R_2, r, u, l) \quad (6)$$

(7)

with

$$r(a) = \begin{cases} r_1(a) \cup r_2(a), & \text{if } a \in A_1 \cap A_2 \\ r_1(a), & \text{if } a \in A_1 \setminus A_2 \\ r_2(a), & \text{if } a \in A_2 \setminus A_1 \end{cases},$$

$$u(a,r) = \begin{cases} u_1(a,r), & \text{if } r \in R_1 \\ u_2(a,r), & \text{if } r \in R_2 \end{cases},$$

and

$$l(a,r) = \begin{cases} l_1(a,r), & \text{if } r \in R_1 \\ l_2(a,r), & \text{if } r \in R_2 \end{cases}.$$

Since the slots (resources) of component heaps are disjoint, l and u are well defined: even though $A_1 \cap A_2 \neq \emptyset$, for any $r \in R$ there is only one $i \in \{1,2\}$, namely such that $r \in R_i$, with $l_i(a,r)$ being defined.

Similarly as in the supervisory control of (logical) automata the purpose of synchronous product is twofold. Firstly, explicitly concurrent heap models (*cf.* concurrent or modular automata) are heap models built by the synchronous product of "local" heap models, whence the interest in studying the properties of synchronous composition of heap models. Secondly, synchronous product is used to describe the action of the supervisor, *i.e.* interaction of the supervisor with the system (*cf.* (Kumar and Heymann, 2000)).

Note that if the above definition is used for control purposes, it is symmetric with respect to both the plant (say $\mathcal{H}_1$) and the controller (say $\mathcal{H}_2$). We then implicitly assume in the above definition that the supervisor is complete, *i.e.* that it never attemps to disable an uncontrollable task. This is always true if all tasks are controllable.

Now (max,+)-linear representation (2), (3), (4) of heap models will be used in study of the morphism matrix of the synchronous product of two heap models. An approach for just in time control of flexible manufacturing systems based on Petri net and heap models, that builds upon the approach of (Menguy, 1997), has been developped in (Al Saba et al., 2006). Our aim is to develop the control theory directly for heap models using synchronous composition of a heap model with its controller (another heap model).

Let us consider the following flexible manufacturing system modelled by Petri net in Figure 1.

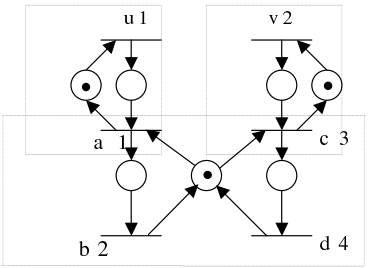


Figure 1: The Petri net model of a flexible manufacturing system.

Note that this 1-safe Petri net is T-timed (timing is associated to transitions) and it can be decomposed into three parts, which are synchronized using shared (synchronization) transitions a and c. Equivalently, each component of this Petri net can be viewed as a separate heap model. The "global" heap model corresponding to the whole timed Petri net is the synchronous product of "local" heap models.

An order relation on $\mathbb{R}_{max}(A)$ will be needed for introduction and study of control problems in Section 5. Let us recall the natural order relation on formal power series from $\mathbb{R}_{max}(A)$. For $s,s' \in \mathbb{R}_{max}(A)$ we put $s \preceq s'$ iff $\forall w \in A^* : s(w) \leq s(w')$, where in the latter inequality usual order in R_{max} that coincides with the natural order is used.

Similarly as a TEG admits a linear representation in the dioid of formal power series $\mathbb{Z}_{max}(\gamma)$ (Baccelli et al., 1992), a heap model admits linear representation in the dioid of formal power series with noncommutative variables from A: $\mathbb{R}_{max}(A)$. As example let us consider the following heap automaton $\mathcal{H}$.

The set of tasks is $A = \{a,b,c,d\}$. The set of resources is $R = \{r_1,r_2,r_3\}$. Let $r(a) = r(b) = \{r_1,r_2\}$ and $r(c) = r(d) = \{r_1,r_3\}$. The lower and upper contours are given by

$$\begin{aligned} l(a,.) &= [0\ \ 0\ -\infty],\ l(b,.) = [0\ \ 0\ -\infty] \\ l(c,.) &= [0\ -\infty\ 0],\ l(d,.) = [0\ -\infty\ 0] \\ u(a,.) &= [0\ \ 1\ -\infty],\ u(b,.) = [2\ \ 0\ -\infty] \\ u(c,.) &= [0\ -\infty\ 3],\ u(d,.) = [4\ -\infty\ 0] \end{aligned}$$

It has been shown in (Gaubert and Mairesse, 1999) that any heap model is a special (max,+)-automaton with the morphism matrix defined in equation (5). The graphical interpretation of the morphism matrix is given in terms of transition weights: $\mu(a)_{ij} = k$ means that there is a transition labelled by $a \in A$ from state i to state j with weight k provided $k \neq -\infty$, in case $k = -\infty$ there is no transition from i to j. Note that the morphism matrix μ of a heap model can be also considered as element of $R_{max}(A)^{|R|\times|R|}$, i.e. $\mu = \oplus_{w\in A^*}\mu(w)w$ by extending the definition of μ from $a \in A$ to $w \in A^*$ using the morphism property

$$\mu(a_1 \ldots a_n) = \mu(a_1) \ldots \mu(a_n).$$

However μ has an important property of being finitely generated, because it is completely determined by its values on A. For this reason we have in fact $\mu^* = (\oplus_{a\in A}\mu(a)a)^*$. Since we are interested in behaviors of heap models that are given in terms of μ^* we abuse the notation and write simply $\mu = \oplus_{a\in A}\mu(a)a$.

The corresponding heap automaton is in Figure 2 below. The state vector is associated to resources of $\mathcal{H}$ variables (formal power series) $x_2,x_1,x_3 \in \mathbb{R}_{max}(A)$ from left to right. We obtain the following

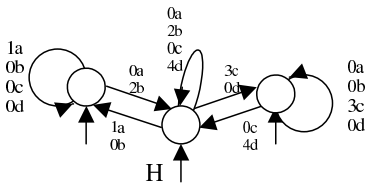


Figure 2: Example of a heap automaton.

equations in dioid $\mathbb{R}_{max}(A)$ endowed with pointwise addition and convolution multiplication:

$$\begin{aligned} x_1 &= x_1 \otimes (0a \oplus 2b \oplus 0c \oplus 4d) \oplus x_2 \otimes (0a \oplus 2b) \\ &\oplus \quad x_3 \otimes (0c \oplus 4d) \oplus e \\ x_2 &= x_1 \otimes (1a \oplus 0b) \oplus x_2 \otimes (1a \oplus 0b \oplus 0c \oplus 0d) \oplus e \\ x_3 &= x_1 \otimes (3c \oplus 0d) \oplus x_3 \otimes (0a \oplus 0b \oplus 3c \oplus 0d) \oplus e \\ y &= x_1 \oplus x_2 \oplus x_3. \end{aligned}$$

The corresponding matrix form is

$$\begin{aligned} x &= x\mu \oplus \alpha \\ y &= x\beta, \end{aligned}$$

with $x = (x_1\ x_2\ x_3)$, $\alpha = (e\ e\ e)$, $\beta = (e\ e\ e)^T$, $\mu =$

$$\begin{pmatrix} 0a \oplus 2b \oplus 0c \oplus 4d & 1a \oplus 0b & 3c \oplus 0d \\ 0a \oplus 2b & 1a \oplus 0b \oplus 0c \oplus 0d & \varepsilon \\ 0c \oplus 4d & \varepsilon & 0a \oplus 0b \oplus 3c \oplus 0d \end{pmatrix}.$$

In general we have the following linear description of (max,+) automata in the dioid $\mathbb{R}_{max}(A)$ of formal power series:

$$x = x\mu \oplus \alpha \quad (8)$$
$$y = x\beta, \quad (9)$$

where $\mu = \bigoplus_{a \in A} \mu(a)a \in \mathbb{R}_{max}(A)$ is also called the morphism matrix.

Recall that the least solution to this equation is $y = \alpha\mu^*\beta$.

4 MORPHISM MATRIX OF SYNCHRONOUS PRODUCT OF HEAP MODELS

Since we introduce supervisory control of heap models using the synchronous product of the plant heap model with the controller heap model, it is important to study properties of the synchronous product. In this section the behavior of a synchronous product of two heap models will be represented in terms of morphism matrix of the synchronous product. According to Definition 3.1, the dimension (here number of resources) of the synchronous product of subsystems is the sum of dimensions of each subsystem. It should be intuitively clear that morphism matrices of synchronous products are block matrices, where the blocks are formed according to dimensions (number of resources) of the component heap models.

In order to simplify the approach we require that $l(a,r) = 0$ whenever $r \in R(a)$, which simply means that pieces are on the ground in all slots. Thus, the upper contour gives information about the duration of tasks for different resources used. We recall at this point that $l(a,r) = -\infty$ for $r \notin R(a)$.

Let $\mathcal{H}_1$ and $\mathcal{H}_2$ be two heap models with morphism matrices denoted by μ_1 and μ_2, respectively. Let ε_{ij}, $i,j = 1,2$ be the rectangular matrices of zeros $(-\infty)$ of dimensions $m_i \times m_j$ and E_i, $i = 1,2$ the square (max,+)-identity matrices of dimensions $m_i \times m_i$. We have the following block form of $\mu_{\mathcal{H}}(a)$, $a \in A$.

Theorem 4.1 *The morphism matrix of $\mathcal{H} = \mathcal{H}_1 \| \mathcal{H}_2$ admits the following decomposition:*

- *If $a \in A_1 \cap A_2$ then:*

$$\mu_{\mathcal{H}}(a) = \begin{bmatrix} \mu_1(a) & \overline{\mu_2}(a) \\ \overline{\mu_1}(a) & \mu_2(a) \end{bmatrix}, \text{ where}$$

$$\overline{\mu_1}(a)_{ij} = \begin{cases} \mu_1(a)_{i'j} : i' \in r_1(a) \text{ arb.}, & \text{if } i \in r_2(a) \\ -\infty, & \text{if } i \notin r_2(a) \end{cases}$$

and

$$\overline{\mu_2}(a)_{ij} = \begin{cases} \mu_2(a)_{i'j} : i' \in r_2(a) \text{ arb.}, & \text{if } i \in r_1(a) \\ -\infty, & \text{if } i \notin r_1(a) \end{cases}$$

- *If $a \in A_1 \setminus A_2$ then :*

$$\mu_{\mathcal{H}}(a) = \begin{bmatrix} \mu_1(a) & \varepsilon_{12} \\ \varepsilon_{21} & E_2 \end{bmatrix}$$

- *If $a \in A_2 \setminus A_1$ then:*

$$\mu_{\mathcal{H}}(a) = \begin{bmatrix} E_1 & \varepsilon_{12} \\ \varepsilon_{21} & \mu_2(a) \end{bmatrix}$$

Proof For $a \in A = A_1 \cup A_2$ three cases must be distinguished. Firstly, $a \in A_1 \cap A_2$ is a shared task. Then for resources r,s from $R_1 \cup R_2$ there are four possibilities depending on whether these are in R_1 or R_2, whence the block form of $\mu_{\mathcal{H}}(a)$. We recall here that $R_1 \cap R_2 = \emptyset$. It is easy to see that in the diagonal blocks the individual morphism matrices appear. Also,

$$\overline{\mu_1}(a)_{ij} = \begin{cases} u(a,j) - l(a,i) = u_1(a,j) \\ \quad \text{if } i \in R_2(a) \text{ and } j \in R_1(a) \\ -\infty \\ \quad \text{if } i \notin R_2(a) \text{ or } j \notin R_1(a) \end{cases}$$

and similarly for $\overline{\mu_2}(a)$. Since $\overline{\mu_1}(a)_{ij}$ equals either $u_1(a,j)$(for $i \in r_2(a)$ and $j \in r_1(a)$) or $-\infty$ otherwise, we can see that the rows corresponding to $i \in r_2(a)$ of $\overline{\mu_1}(a)$, i.e. $\overline{\mu_1}(a)(i,.)$, are the same as the rows $\mu_1(a)(i',.)$ for $i' \in r_1(a)$, which are all the same, *i.e.* an arbitrary one can be taken. Thus, the corresponding entry does not depend on j anymore. Hence, the morphism matrix of the composed heap $\mu_{\mathcal{H}}(a)$ has the claimed form. In the much easier situation when $a \in A_1 \setminus A_2$ it is sufficient to notice that no resource from R_2 is used by a. It is easily seen that $\mu_{\mathcal{H}}(a)$ has again the claimed form. Finally, the case $a \in A_2 \setminus A_1$ is symmetric.

Theorem 4.1 can be generalized to $n \in \mathbb{N}$, where morphism matrix of synchronous product are matrices with $n \times n$ blocks. This is useful for decentralized control, but in this paper we only need synchronous product of the system with its controller.

We recall at this point that

$$\mu_H = \bigoplus_{a \in A} \mu_H(a) \otimes a \in \mathbb{R}_{max}(A).$$

The algebraization of synchronous product presented in this section will be useful for control purposes in the next section.

5 SUPERVISORY CONTROL OF HEAP MODELS

In this section supervisory control of heap models is studied. The aim is to satisfy a behavioral specification given by a formal power series. The closed-loop system is represented by parallel composition (synchronous product) of the plant with a supervisor to be found, which is itself represented by a heap model.

In general a supervisor acts on both timing and logical properties of the plant's behavior under supervision. Since heap models are special (max,+)-automata there are two aspects of supervision: disabling and delaying of tasks (events). Here we are only interested in delaying the different tasks which is similar to control of TEG in the maxplus algebra, where input transitions are added in order to delay the timed behavior of a TEG.

Now we show how Theorem 4.1 can be used for control of heap models. The synchronous product of heap models $G = (A_g, R_g, r_g, l_g, u_g)$ and $C = (A_c, R_c, r_c, l_c, u_c)$ of dimensions m and n corresponds to the controlled (closed-loop) system. The event alphabet of the controlled system is denoted by A. According to Definition 3.1, we have $A = A_g \cup A_c$. Let us denote the morphism matrices of G and C by μ_g and μ_c, respectively. Now let us return to the description of behaviors of heap models in the dioid of formal power series $\mathbb{R}_{max}(A)$. The vector of formal power series from $\mathbb{R}_{max}(A)$ associated to generalized dater functions $x_{\mathcal{G}\|\mathcal{C}} : A^* \to \mathbb{R}_{max}^{m \times n}$ satisfies the following equations:

$$x_{\mathcal{G}\|\mathcal{C}} = x_{\mathcal{G}\|\mathcal{C}} \mu_{\mathcal{G}\|\mathcal{C}} \oplus \alpha, \quad (10)$$

$$y_{\mathcal{G}\|\mathcal{C}} = x_{\mathcal{G}\|\mathcal{C}} \beta, \quad (11)$$

where $\mu_{\mathcal{G}\|\mathcal{C}}$ is the morphism matrix of $\mathcal{G}\|\mathcal{C}$, α and β are row, resp. column, vectors of 0's of dimension $m+n$. The structure of the morphism matrix described in Theorem 4.1 is now used for control purposes. According to Theorem 2.1 the greatest solutions to equations (10) and (11) are

$$x_{\mathcal{G}\|\mathcal{C}} = \alpha \mu^*_{\mathcal{G}\|\mathcal{C}}, \quad (12)$$

$$y_{\mathcal{G}\|\mathcal{C}} = \alpha \mu^*_{\mathcal{G}\|\mathcal{C}} \beta, \quad (13)$$

whence an interest in studying properties of $\mu^*_{\mathcal{G}\|\mathcal{C}}$.

Given a specification behavior (e.g. language or formal power series), the goal in supervisory control of DES is to find a supervisor that achieves this specification as the behavior of the controlled system. In a first approach we assume, similarly as in control of TEG, that the structure of the controller is given, which means here that the controller heap model only delays task executions of the plant. This is done by the choice of upper contour functions (*i.e.* duration of controller's tasks) from $\mu_c(u)$. The delaying effect of the controller is naturally realized *via* its tasks (transitions of the corresponding heap automaton) shared with the plant heap.

The morphism matrix of the composed system is given by

$$\mu_{\mathcal{C}\|\mathcal{G}} = \bigoplus_{u \in A_c \setminus A_g} \mu_{\mathcal{C}\|\mathcal{G}}(u) u \oplus \bigoplus_{a \in A_c \cap A_g} \mu_{\mathcal{C}\|\mathcal{G}}(a) a \oplus \bigoplus_{a \in A_g \setminus A_c} \mu_{\mathcal{C}\|\mathcal{G}}(a) a.$$

In order to simplify the approach our attention is from now on limited to the case $A_c = A_g$, which is a standard assumption in the supervisory control with complete observations. Since state vector in equation (8) is associated to resources, it can be written as $x_{\mathcal{C}\|\mathcal{G}} = (x \ u)$, where the first component corresponds to the (uncontrolled) plant heap and the second to the controller heap. Owing to Theorem 4.1 we have:

$$(x \ u) = (x \ u) \bigoplus_{a \in A} \begin{bmatrix} H(a)a & \bar{F}(a)a \\ \bar{H}(a)a & F(a)a \end{bmatrix} \oplus (\alpha_1, \alpha_2),$$

where α_1 and α_2 are vectors of zeros of corresponding dimensions, $\mu_g(a)$ is for convenience denoted by

$\underline{H}(a)$, $\overline{\mu_g}(a)$ by $\bar{H}(a)$, $\mu_c(a)$ by $F(a)$, and $\overline{\mu_c}(a)$ by $\bar{F}(a)$. This can be written as

$$(x\ \ u) = (x\ \ u)\begin{bmatrix} A & \bar{F} \\ \bar{A} & F \end{bmatrix} \oplus \alpha,$$

where $H = \bigoplus_{a\in A} H(a)a$ and similarly for $F, \bar{F}$, and $\bar{H}$. Hence,

$$x = xH \oplus u\bar{H} \oplus \alpha_1$$
$$u = x\bar{F} \oplus uF \oplus \alpha_2.$$

Theorem 2.1 yields $u = (x\bar{F} \oplus \alpha_2)F^*$ as the least solution of the second equation, which substituted into the first equation leads to

$$x = xH \oplus \left[(x\bar{F} \oplus \alpha_2)F^*\right]\bar{H} \oplus \alpha_1.$$

The least solution is given by

$$x = (\alpha_2 F^* \bar{H} \oplus \alpha_1)(H \oplus \bar{F}F^*\bar{H})^*. \qquad (14)$$

The last equation can be viewed as the expression of the closed-loop system using the feedback given by F. Notice that unlike classical control theory the control variables have their inner dynamics, which is caused by adopting a supervisory control approach, where a controller is itself a dynamical system of the same kind as the uncontrolled system. Therefore, our $\bar{F}$, which plays the role of feedback mapping, is determined by inner dynamics of the controller given by its morphism matrix F.

There is a strong analogy with the feedback approach for control of TEG, see (Cottenceau et al., 2001), which should not be surprising, because supervisory control (realized here by synchronous product) is based on a feedback control architecture. In supervisory control the control specification (as counterpart of reference output from control of TEG using dioid algebras) are given in terms of behaviors of (max,+) automata (*i.e.* formal power series). In fact, for a reference output y_{ref} we are interested in the greatest F such that

$$y \leq y_{ref},$$

thus,

$$(\alpha_2 F^* \bar{H} \oplus \alpha_1)(H \oplus \bar{F}F^*\bar{H})^*\beta \leq y_{ref}. \qquad (15)$$

We obtain from Lemma 2.2 that

$$(H \oplus \bar{F}F^*\bar{H})^* = H^*(\bar{F}F^*\bar{H}H^*)^*, \text{i.e.}$$

inequality (15) becomes

$$(\alpha_2 F^* \bar{H} \oplus \alpha_1)H^*(\bar{F}F^*\bar{H}H^*)^*\beta \leq y_{ref} \qquad (16)$$

Note that while $\otimes$ and $\oplus$ are lower semicontinuous and residuated, the Kleene star is not in general, but only when the image is suitably constrained (in which case it is trivially residuated with identity as the corresponding residuated mapping). Moreover, as follows from Theorem 4.1, $\bar{F}$ can be simply expressed using F. Hence, there is a hope that at least in some special cases residuation theory (see Section 2) can be applied to obtain the greatest series F corresponding to the "controller part" of the morphism matrix $\mu_{\mathcal{G}\|\mathcal{C}}$ such that $y_{\mathcal{G}\|\mathcal{C}}$ satisfies a given specification (e.g. is less than or equal to a given reference output series y_{ref}). In fact we obtain from the above inequality the greatest $\bar{F}F^*$ such that inequality (16) is satisfied. Thus it is not an easy problem. The situation is much simpler in case $\underline{F} = F$ and $\bar{H} = H$. This is satisfied if we assume that the controller heap model has the same number of resources as the uncontrolled heap model and the logical structure of the controller (given by $r_c : A \to Pwr(R_c)$) mimicks the structure of the plant, formally there exists an isomorphism between R_c and R_g such that r_c and r_g are equal up to this isomorphism. In terms of Petri nets this can be interpreted as having a controller net with the same net topology (*i.e.* logical structure as the uncontrolled net), i.e. in the closed-loop system there are always parallel places of the controller corresponding to places of the uncontrolled net. The role of the controller is only to act on the system through holding times of the controller's places that correct the holding times of the places in the original net. Because of the fixed parallel structure of the controller it is clear that the controller can in this case only delay the firing of the transitions, which are all shared by the system and the controller.

It is easy to check that Theorem 4.1 in such a case gives $\bar{F} = F$ as well as $\bar{H} = H$ and $\alpha_1 = \alpha_2 = \alpha$, row vector of zeros of dimension n. Hence, inequality (15) becomes $\alpha(F^*H \oplus E)(H \oplus \bar{F}F^*H)^*\beta \leq y_{ref}$, where E is the identity matrix. An easy calculation yields $(H \oplus \bar{F}F^*H) = (E \oplus F^+)H = F^*H$, hence

$$(F^*H \oplus E)(F^*H)^* = (F^*H)^+ \oplus (F^*H)^* = (F^*H)^*.$$

Thus, $y = \alpha(F^*H)^*\beta \leq y_{ref}$, *i.e.* the problem is to find the greatest F such that

$$(F^*H)^* \leq \alpha \backslash y_{ref} \not/ \beta.$$

Since the Kleene star is not a residuated mapping in general, such a problem has only a solution if $\alpha \backslash y_{ref} \not/ \beta$, playing the role of reference model G_{ref} from (Cottenceau et al., 2001) is of a special form to be studied. Let us notice that $H \geq E$, which follows from the form of morphism matrix and the usual assumption that any resource of the system is used by at least one task: $\forall r \in R_g\ \exists a \in A$ such that $r \in r_g(A)$. The following Lemma is useful.

Lemma 5.1 *If $H \geq E$ then for any $B \in R_{max}(A)^{n\times n}$ every solution of $(X^*H)^* \leq B$ is a solution of $H(X^*H)^* \leq B$ and vice versa.*

Proof If X a solution of $(X^*H)^* \leq B$, then $(X^*H)^* = E \oplus (X^*H)^+ \leq B$, hence also $(X^*H)^+ \leq B$. Therefore,

$$H(X^*H)^* \leq X^*H(X^*H)^* = (X^*H)^+ \leq B,$$

where the first inequality follows from isotony of multiplication and $E \leq X^*$. Conversely, if X is a solution of $H(X^*H)^* \leq B$, then $(X^*H)^* = E \otimes (X^*H)^* \leq H(X^*H)^* \leq B$ as follows from isotony of multiplication and the assumption that $E \leq H$.

Using Lemma 5.1 our problem is to find the greatest solution in F of

$$H(F^*H)^* \leq \alpha \backslash y_{ref} \not\backslash \beta.$$

It follows from Lemma 2.2 that

$$H(F^*H)^* = (H \oplus F)^* = H^*(FH^*)^*,$$

thus we get formally the same problem as the one solved in (Cottenceau et al., 2001) with H^* playing the role of transfer function H in the TEG setting. The following result adapted from (Cottenceau et al., 2001), *Proposition 3*, is useful: If there exists $D \in R_{max}(A)$ such that $\alpha \backslash y_{ref} \not\backslash \beta = H^*D^*$ or there exists $D' \in R_{max}(A)$ such that $\alpha \backslash y_{ref} \not\backslash \beta = D'^*H^*$ then there exists the greatest F such that $H^*(FH^*)^* \leq \alpha \backslash y_{ref} \not\backslash \beta$, namely

$$F^{opt} = H^* \backslash [\alpha \backslash y_{ref} \not\backslash \beta] \not/ H^* = \alpha H^* \backslash y_{ref} \not\backslash H^* \beta.$$

In the special case we have restricted attention to, our methods yields the gretest feedback such that timing specification given by y_{ref} is satisfied, provided y_{ref} is of one of the special forms. In the special case of a controller with fixed logical structure only timed behavior is under control.

If we are interested in manufacturing systems, where specificatons are given in terms of Petri nets, the reference output is not typically required to be met for all sequences of tasks, but only those having a real interpretation. These are given by the correponding (logical) Petri net language, say L. Thus, the problem is to find the greatest F, such that

$$\alpha H^*(FH^*)^* \beta \; char(L) \leq y_{ref} \; char(L),$$

where $char(L) = \bigoplus_{w \in L} e.w$ is the series with Boolean coefficients, *i.e.* the formal series of language L. Let us recall (Gaubert and Mairesse, 1999) that such a restricton is formally realized by the tensor product (residuable operation) of the heap automaton with the logical (marking) automaton recognizing the Petri net language L, which is compatible with Theorem 4.1 of (Komenda et al., 2007).

Note that specifications based on (multivariable) formal power series are not easy to obtain in many practical problems, in particular those coming from production systems, often represented by Petri nets. In fact, given a reference output series amounts to solve a scheduling problem. A formal power series specification is not given, but it is to be found: e.g. using Jackson rule (Jackson, 1955).

6 CONCLUSION

It has been shown how methods of dioid algebras can be used in supervisory control of heap models. We have proposed a synchronous product of heap models. The structure of the morphism matrix of synchronous product of two heap models is derived and applied to control of heap models.

The present reseach is a very first step in control of heap automata. Sharing of resources is only allowed inside component heap models. Of potential interest is supervisory control with partial controllability, partial observations, and decentralized control of heap automata.

ACKNOWLEDGEMENTS

Partial financial support of Université d'Angers, of the Grant Agency GA AV No. KJB100190609, and of the Academy of Sciences of the Czech Republic, Institutional Research Plan No. AV0Z10190503 is gratefully acknowledged.

REFERENCES

M. Al Saba, J.L. Boimond, and S. Lahaye. *On just in time control of flexible manufacturing systems via dioid algebra.* Proceedings of INCOM'06, Saint-Etienne, France, vol.2, pp. 137-142, 2006.

F. Baccelli, G. Cohen, G.J. Olsder and J.P.Quadrat (1992). *Synchronization and linearity. An algebra for discrete event systems.* New York, Wiley.

B. Cottenceau, L. Hardouin, J.L. Boimond, and J.L. Ferrier. *Model Reference Control for Timed Event Graphs in Dioid*, Automatica, vol. 37, pp. 1451-1458, 2001.

S. Gaubert. *Théorie des systèmes linéaires dans les dioïdes*. Thèse de doctorat, Ecole des Mines de Paris, 1992.

S. Gaubert. *Performance evaluation of (max,+) automata*, IEEE Trans. on Automatic Control, 40(12), pp. 2014-2025, 1995.

S. Gaubert and J. Mairesse. *Task resource models and (max,+) automata*, In J. Gunawardena, Editor: Idempotency. Cambridge University Press, 1997.

S. Gaubert and J. Mairesse. *Modeling and analysis of timed Petri nets using heaps of pieces.* IEEE Trans. on Automatic Control, 44(4): 683-698, 1999.

J.R. Jackson. *Scheduling a Production Line to Minimize Maximum Tardiness.* Research report 43. University of California Los Angeles. Management Science Research Project.

J.Komenda, M.Al Saba, and J.L. Boimond. *Supervisory Control of Maxplus Automata: Quantitative Aspects.* In Proceedings ECC 2007, Kos (Greece).

R. Kumar, M. Heymann. Masked prioritized synchronization for interaction and control of discrete-event systems, IEEE Transaction Automatic Control 45, 1970-1982, 2000.

F. Lin and W.M. Wonham, On Observability of Discrete-Event Systems, *Information Sciences*, 44: 173-198, 1

E. Menguy. *Contribution à la commande des systèmes linaires dans les diodes*. Thèse de doctorat, Université d'Angers, 1997.

P.J. Ramadge and W.M. Wonham. The Control of Discrete-Event Systems. *Proc. IEEE*, 77:81-98, 1989.

J. Sifakis and S. Yovine. Compositional specification of timed systems. Proceedings of the 13th Symp. on Theoretical Aspects of Computer Science, STACS'96, pp. 347-359, 1996. LNCS 1046.

ON THE USE OF AUTOMATED GUIDED VEHICLES IN FLEXIBLE MANUFACTURING SYSTEMS

Samia Maza* and Pierre Castagna**
* *Centre de Recherche en Automatique de Nancy, UMR 7039, CNRS-Nancy university*
2, avenue de la Forêt de Haye, 54506 Vandoeuvre lès Nancy, France
** *Institut de Recherche en Communications et Cybernétique de Nantes, UMR CNRS 6597*
1, rue de La Noë, BP 92101 Nantes Cedex 3, France
samia.maza@ensem.inpl-nancy.fr, pierre.castagna@univ-nantes.fr

Keywords: Manufacturing systems, automated guided vehicles, conflicts, routing, traffic management.

Abstract: In manufacturing systems, material transport plays a key role for production process efficiency. Because of their advantages over other material handling systems such as conveyors and robots, AGVs are widely used in flexible manufacturing systems. The scheduling of several AGVs in a non-conflicting manner is a complicated problem, especially when the AGV system is bi-directional. In fact, many undesirable situations may arise such as deadlocks and head-on conflicts if no efficient control policy is used to prevent them. This paper presents the key issues to be addressed to efficiently employ these devices, and deal particularly with the traffic management problem.

1 INTRODUCTION

Automatic Guided Vehicle Systems (AGVS) are one of the most exciting and dynamic areas in material handling today. But AGVS is really not so new. Fifty years ago when AGVS was invented it was then called driverless systems. Through the years, advances in electronics have led to advances in guided vehicles. Technological developments may have given AGVS more flexibility and capability, but market acceptance has really given AGVS the application variety to allow it to expand into the standard accepted material handling method it is today.

Automated guided vehicles (AGVs) are material handling devices used to transport products and goods among the workstations and storage areas of a manufacturing system. The basic functions of an AGVS are:

- Navigation and Guidance allow the vehicle to follow a predetermined route which is optimized for the material flow pattern of a given application
- Routing is the vehicle's ability to make decisions along the guidance path in order to select optimum routes to specific destinations
- Traffic Management is a system or vehicle ability to avoid collisions with other vehicles while at the same time maximizing vehicle flow and therefore load movement throughout the system.
- Load Transfer is the pickup and delivery method for an AGVS system, which may be simple or integrated with other subsystems.
- System Management is the method of system control used to dictate system operation.

The goal of this paper is to present some interesting problems related to the use of AGV systems and a short overview of papers dealing with those problems especially the routing problem. We will present our research works and results concerning the routing of bi-directional AGV Systems.

In the second section, the key issues to be adressed to efficiently use the AGVs are presented. The approaches we have developed for the conflict free routing of bi-directional AGVs will be briefly presented in the third section. The simulation study is presented in section four and the study we have made for the compact disc manufacturer is presented in section five. Section six is devoted to the conclusions.

2 DESIGN AND CONTROL OF AN AGVS

An AGV system is a set of a cooperative driver-less vehicles moving on the same manufacturing floor and coordinated by a control system.

For a successful deployment of an AGVS, the following key issues should be addressed (Reveliotis, 2000):

2.1 The Flow Path Design

The manufacturing floor is specified by a set of physical or virtual guide-paths. If the AGVs are allowed to move only according to one direction, they are called unidirectional, otherwise, i.e., if they can move into the two directions, they are said to be bi-directional. It has been shown that the bi-directional AGVS can improve considerably the performances of a manufacturing system (Egbelu and Tanchoco, 1986). The design of the guide-path is an important problem, i.e., the choice of the guide-path configuration in order to minimise the travelled distances. Many research works deal with such a problem, the others deal with the determination of the guide-path's lanes direction for unidirectional AGVSs (for example Gaskins and Tanchoco, 1987). Such a problem is generally formalised as an integer programming problem with distance as criterion to optimise. The most known configurations are the simple loop, multiple loops, tandem and conventional configurations.

2.2 The Fleet Size and Vehicle's Capacity Determination

The AGVs can be classified according to their load capacity into two categories: single and multiple load vehicles. And one of the important problems to be addressed when designing AGVS is the determination of the AGVs number and their loading capacity. The research works which deal with such problem are numerous and can be classified into three categories:

- Analytical methods;
- Simulating methods;
- Hybrid or mixed methods.

For example in (Egbelu and Tanchoco, 1987), four analytical methods are proposed to determine the minimum fleet size to satisfy the production needs. These methods were tested for various dispatching rules. Beamon and Chen (Beamon and Chen, 1998a) reproach the traditional methods to not consider the system reliability when determining the optimal fleet size. To consider the reliability of the AGVs and the guide-path intersections, they include in their proposed approach, the vehicles and intersections failure rates in order to calculate the minimum number of AGVs needed. Beamon and Deshpande (Beamon and Deshpande, 1998b) proposed an approach to jointly optimise the fleet size the vehicle's load capacity (i.e., the load batch size). The objective is to make the better trade-off between those two criteria. Indeed, more great is the load size; shorter will be the total travelled distances, since one vehicle will make only one displacement with many loads. However, the loading and unloading time will be greater than for a single load. In the same manner, when there is a great number of AGVs, the system performances will be improved until the optimal number. Beyond this optimal number, the performances will be degraded since a great number of vehicles increases the traffic congestion and deadlocks.

- In (Castagna and Maza, 2004) we proposed a simulation approach to determine the optimal fleet size given a production horizon T, the manufacturing ranges, the production rate, and the guide-path.

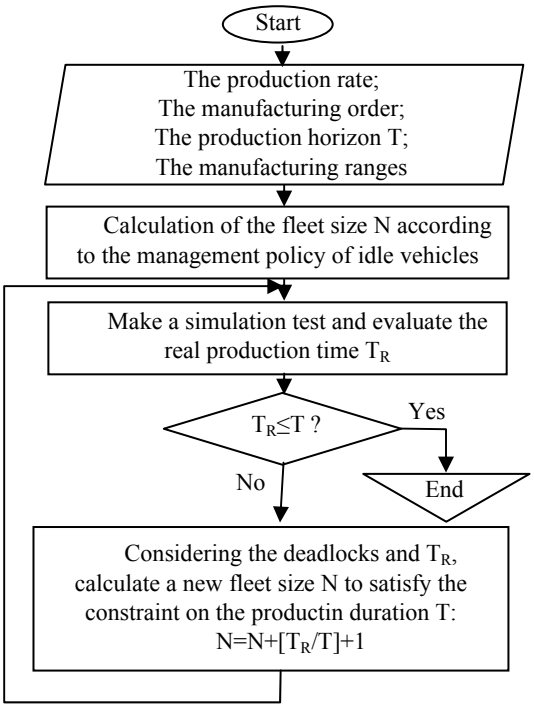


Figure 1: the fleet size calculation procedure.

The basic idea of this approach is:
In the first step, the number of AGVs needed to realise the specified number of products into the specified makespan T is calculated.

In the second step, a simulation study is conducted to test the calculated number; and to refine it by considering the achieved performances (see figure 1). Indeed, when the fleet size is calculated in the first step, deadlocks are not considered. The simulation allows the determination of the real production time T_R needed to accomplish all the transportation missions. This new makespan is used to determine the AGVs fleet size once again and is tested by simulation. The optimal fleet size is obtained when all the transportation tasks are effectively achieved in the specified production horizon T.

This study was made for various management policies of idle AGVs and was compared to the analytical approach proposed in (Egbelu and Tanchoco, 1987).

2.3 The Dispatching Problem

To achieve a product or a job, it has to be routed on several stations of the manufacturing system to undergo some transformation operations. These are those transitions which introduce the problem of vehicle's task assignment.

Indeed, when an operation is completed on one workstation, the product makes a request for a vehicle to be transported to the next station.

The dispatching problem consists in choosing one request among several ones in a standby state, and choosing one vehicle to be affected to that request. This vehicle should be able to make the resulting displacement.

The dispatching problem was developed in many research papers, but their number is still smaller than the one of the papers dealing with the scheduling problem without transportation resources. For example Blazewicz et al (Blazewicz et al, 1991) propose an approach to search for a scheduling that jointly considers the jobs and vehicles. Egbelu and Tanchoco (Egbelu and Tanchoco, 1984) describe the major vehicles' dispatching rules for two special cases: (a) when there is only one transporting request and many idle vehicles, and (b) when there is only one idle AGV and many jobs requesting a vehicle.

Other dispatching cases were considered in (Albert, 1998).

2.4 The AGVs Routing and Traffic Management Problem

The aim of routing AGVs is to find an optimal (e.g. shortest possible time path) and feasible route for every single AGV.

Actually, the routing decision includes three aspects. Firstly, it should detect whether there *exists* a route which could lead the vehicle from its origin to the destination. Secondly, the route selected for an AGV must be feasible, i.e., the route must be congestion-, conflict- and deadlock-free (Taghaboni and Tanchoco, 1995), etc. Thirdly, the route must be optimal or at least partially optimal, e.g. minimize idling runs of vehicles.

Indeed, AGVS are the seat of a great number of undesirable situations, in particular when they are bi-directional. Situations like conflicts and deadlocks. Conflicts occur when for example two AGVs are attempting to travel one lane at the same direction but at different speeds, or into opposite directions. Other conflicts occur when several vehicles attempt to cross one intersection at the same time (Figure 2.)

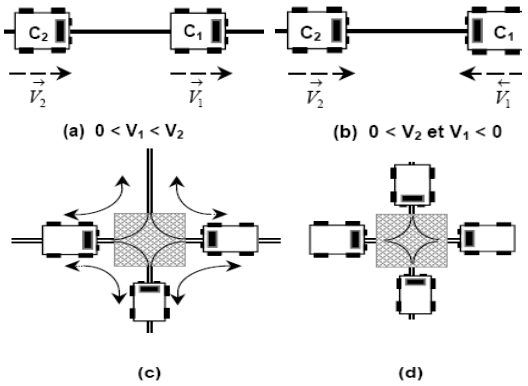


Figure 2: Example of conflicts between AGVs.

Deadlock is a well known problem in the resource allocation systems and technological areas such as computer operating systems, transportation and automated manufacturing systems (Lawley and Reveliotis, 1999).

A resource allocation system (RAS) consists of a finite set of resources that must be allocated to competing processes. The processes enter the system, request, acquire, use, and release their required resources, and then exit the system.

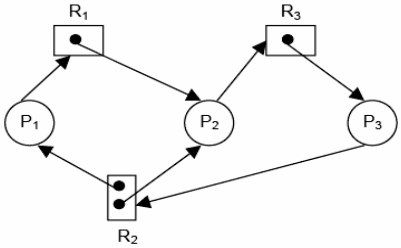


Figure 3: Example of deadlock situation in a resource allocation graph for a RAS of 3 processes and 3 resources, where R_2 is of capacity 2.

Many types of RASs are prone to deadlock, an insidious halting condition in which there exists a set of processes with every process in the set awaiting the allocation of resources held by other processes in the set (Figure 3).

Well known strategies for handling deadlock are (1) prevention, (2) detection-resolution, and (3) avoidance. Prevention restrains the request structure of processes so that deadlock is impossible. Because it limits process concurrency, prevention tends to be overly restrictive and typically achieves poor resource utilization.

Detection-resolution approaches allow deadlock to occur and then concentrate on expedient resolution. This approach achieves the greatest flexibility in resource allocation at the cost of system stoppage and resolution procedures, which may involve aborting processes or the time consuming transport and reshuffling of physical entities. Avoidance uses current state information along with knowledge of process request and release structures to restrain the way resources are allocated so that deadlock never occurs. Avoidance achieves a middle ground in terms of allocation flexibility, being more flexible than prevention but less flexible than detection. It does not incur the cost of system stoppage and resolution and thus is the preferred method when the incremental increase in allocation flexibility does not merit the cost of allowing deadlock to occur. Dijskstra was pioneer in that field and proposed a polynomial algorithm, known as the banker algorithm, to resolve a sequential resource allocation problem. A more complete discussion of fundamental deadlock concepts can be found in most books on computer operating systems, for example see (Silberschatz and Peterson, 1991).

These few last years, many research works were conducted to avoid deadlocks in automated manufacturing systems, for example (Pia Fanti, 1997) and (Reveliotis, 1996).

An AGV system can be considered as a resource allocation system, where the processes are the AGVs and the resources are intersections and lanes of the guide path. The problem of deadlock and conflict free routing in AGVS will be developed in the next section.

3 CONFLICT FREE ROUTING OF BI-DIRECTIONAL AGVS

In our research work, we were especially interested in conflict-free routing of bi-directional AGVS.

Several routing approaches were proposed in the literature and can be classified into two categories:

(1) Predictive or planning methods: here the conflicts are predicted off-line and vehicles' paths are planned to avoid these conflicts and to minimise a performance criterion see for example (Krishnamurthy et al, 1993) and (Oboth et al, 1998).

(2) Reactive or dynamic methods: here, an AGV path is not planned and routing decisions are made in a real time manner according to system's state. Such methods are always qualified as zone dynamic control methods, since the guide path is divided into non overlapping zones considered as non-sharable resources; see for example (Reveliotis, 2000), and (Branislav, 2002).

The advantage of the first category of methods is that the system's performances (like the makespan or travel time of AGVs) are a priori considered and optimised. However, unlike the reactive methods, the planning approaches are sensitive to perturbations since the scheduling is made in a deterministic way. Dynamic methods are very robust but do not consider the performances optimisation beyond a short horizon.

In order to have the advantages of the types of methods, we proposed a new approach in (Maza ad Castagna, 2005a, 2005b) based on a planning method proposed in (Kim and Tanchoco, 1991).

This planning method is based on an algorithm called *cfstp* (i.e. conflict free shortest time procedure), which calculates for each AGV, having a transporting mission the fastest route, considering the traffic status, to reach its destination without conflicts. To this end, intersections and characteristic point of the guide path are modelled by square areas, called nodes, which are considered as non sharable resources. When an AGV moves on this guide path, it reserves some nodes for a while (see figure 4). This duration is called reserved time window (noted r_n for node n), where the node is exclusively reserved by that vehicle. Other time intervals where nodes are free are called free time windows (noted f_m for node m).

The *cfstp* calculates the shortest path on a directed time windows graph, in which the vertices represent the free time windows and the links model the reachability between these time windows. The ability to reach a time window from another one is established by calling another algorithm called *the reachability test procedure*. For two free time windows f_n^p and f_m^q associated respectively to the nodes n and m, this last procedure makes the following reachability tests between them:

(1) Check for space feasibility, i.e., the existence of a physical link relating m to n.
(2) Check for time feasibility, i.e., the node m is reachable from the node n within its free time window f_m^q.
(3) Check for potential conflicts.

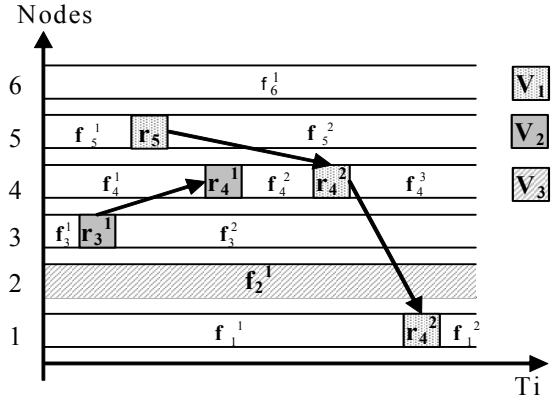


Figure 4: Example of time-windows table.

— r_i^j is the j^{th} reserved time window of node i
— f_l^k is the k^{th} free time window of node l.

When a mission is possible, the *cfstp* delivers for its assigned AGV a set of nodes to be visited and the arrival and exit times to those nodes to avoid conflicts and minimise the travel time. As said before, such method is sensitive to perturbations.

There are two types of contingencies: temporary and permanent. We consider only the first type, such as a slowing down in front of a fixed or a moving obstacle, or a temporary stop on a lane or a node to charge the battery, etc. In that case, the scheduled arrival and exit times will not be respected and consequently, there is no security guarantee for the AGVs since collisions can occur.

To ensure the reliability of an AGVS in the presence of interruptions while maintaining the scheduled trajectories, a control architecture was proposed in (Maza and Castagna, 2001, 2005a).

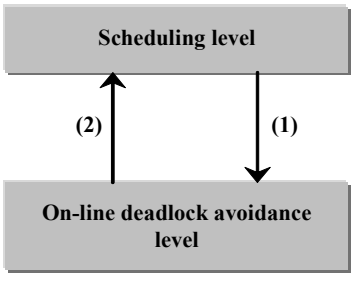


Figure 5: The AGV control architecture.

(1) the scheduling level delivers for each node i, an ordered list, O_i, of AGVs having to cross it in a growing order of their arrival dates.
(2) the deadlock avoidance level operates in presence of contingencies by respecting the predicted node's crossing order (*RVWA*) or by re-ordering the AGVs (*RVRAA* or *RVDA*). It informs the 1st level about the current changes.

Indeed, a second level of real-time control was added to the AGVs scheduling level which uses the *cfstp*, in order to avoid deadlocks and conflicts when needed (see Figure 5). First, the AGVs are scheduled on the nodes of the guide-path in a non conflicting manner while optimising the mission's duration. Then, the scheduled entry times to each node are used to establish for each AGV, its own priority to cross these nodes.

Three polynomial algorithms were proposed. The first one based on static priorities, called *RVWA* (or *Robust Vehicle Waiting Algorithm*). *RVWA* is based on a theorem that says that if each AGV respects its node crossing order, the property of non-conflict is conserved.

The second algorithm based on dynamic priorities, called *RVRAA* (or *Robust Vehicle Routing Ahead Algorithm*) allows the rescheduling of the AGVs on some nodes in order to improve the *RVWA* which always induces unnecessary waiting of vehicles to respect their crossing priorities (Maza and Castagna, 2005a). The basic idea of *RVRAA* is to give the AGV V which calls the algorithm the greatest priority on some specified path [N,M], where N is the node where V calls *RVRAA*, and M is the node where V has the highest priority.

The third algorithm called *RVDA* (or *Robust Vehicle Delaying Algorithm*), also based on dynamic priorities, penalises the AGV which is late, say U, in front of some other AGVs on a path [N,M]. N is the node where the algorithm is called, and M is a node which is calculated by the algorithm to insure that the system will never reach an unsafe state (Maza and Castagna, 2005b). An AGV state is called *unsafe* if it can conduct the AGV system to a deadlock state, i.e., it satisfies the necessary condition for the occurrence of conflicts.
These three algorithms were tested and compared in a simulation study which is developed in the next section.

We have developed another approach for the reactive conflict-free routing of the AGVs based on multi-agent systems (Breton et al, 2006). The main idea of this approach is to consider an AGV as a reactive agent, whose goal is to reach a predefined destination node without conflict with the moving AGVs. In order to design the AGV-agent, the

Cassiopea Multi-Agent System (MAS) design methodology is used (Collinot and Droguoul, 1996).

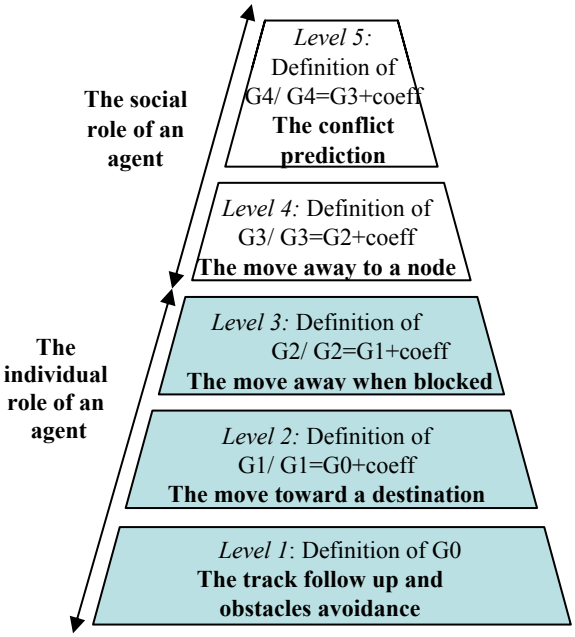


Figure 6: The application of the Cassiopea methodology to design an AGV-agent of a deadlock-free AGV system.

This methodology defines an agent in five incremental layers, considering the agent's different roles. This incremental construction has the advantage of satisfying the principle of parsimony, i.e., the definition of the agent can be stopped as soon as the system completes its desired function.
The steps followed to design an AGV-agent are given in figure 6. A gradient G_j is calculated in each step j to meet some specification (for example a track follow-up). An AGV-agent will move to minimise this gradient, i.e. according to the direction where the gradient is minimal. This approach was also compared in a simulation study to the predictive approach described before. For more details see (Breton et al, 2006).

4 SIMULATION OF BI-DIRECTIONAL AGV SYSTEMS

To test the various approaches described before, we have used the ARENA software to develop a new template panel, which allows us to model the routing of bi-directional AGVs. This function is actually not included in ARENA package. Our template also allows the modelling of real AGVSs, subject to contingencies. This template panel is well described in (Maza and Castagna, 2005a).

To check for the efficiency of our algorithms *RVWA*, *RVRAA* and *RVDA*, we made some simulation tests using our template. The AGVS under study is composed of bi-directional mesh-like guide-path of 45 nodes and 60 links and a fleet of 8 AGVs. Each simulation essay is a sequence of at least 10 replications. In one replication, each AGV has to realise a set of 100 missions randomly generated. To approach reality, random failures of AGVs are generated in the simulation model. They are characterised by two parameters: the failure rate τ and the mean time between failures MTBF.
Different simulations were done with various system parameters in order to compare these algorithms and bring out the situations where the use of one algorithm is more appropriate than another (Maza and Castagna, 2005b).

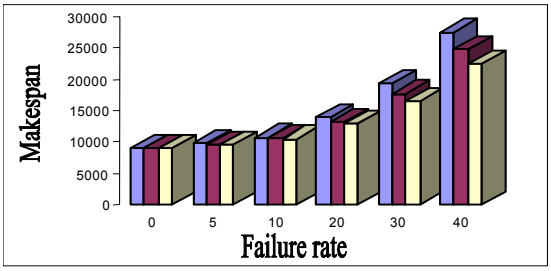


Figure 7: The makespan evolution according to the failure rate (τ=0, 5%, 10%, 20%, 30%, 40%).

For example, by varying the failure rate and fixing other parameters, it can be concluded that more the failure rate is important, better will be the makespan achieved by the algorithms *RVRAA* and *RVDA* and that the *RVDA* gives the best results (Figure 7. This can be explained by the fact that the *RVDA* algorithm is more permissive than the two other algorithms. More simulation results are available in (Maza and Castagna, 2005b).

5 AN INDUSTRIAL APPLICATION

We briefly present here an example of a simulation study we have done for an industrialist MPO who wanted to implement an AGV system. The goal of this study was the prediction of the stores evolution according to various AGVs management policies. For more than fifty years now, MPO has been an expert in the manufacture and replication of pre-recorded media. Every year, 600 millions discs and

150 millions printings elements come out of the MPO's plants.

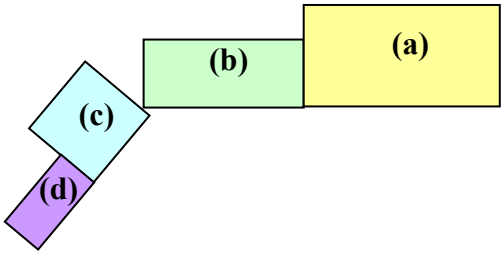


Figure 8: The MPO Production facilities.

This simulation project took place in the production plant of Averton (France). The production plant is composed of four workshops (Figure 8):

The pressing shop: the CDs and DVDs are obtained by injection of fluidized polycarbonate in a press mold. Then, a fine layer of aluminum is deposited in a vacuum, by pulverization, onto the surface of the disc (Figure 8 (a)).

The printing shop: two printing principles are utilized. The first one is silkscreen printing. This technique consists of printing by means of a cloth frame, which favors the flat decorations. The second one is offset printing, by means of linked rolls which is suitable for photos and illustrations in several hues (Figure 8 (b)).

The packaging shop: the discs are packaged in their boxes (Figure 8 (c)).

The storage zone: the store is utilized both to store raw materials and to store the CDs waiting for package (Figure 8 (d)).

All the transports of goods between the workshops are realized using an AGV system.
The particularity of this system is that one AGV is used for towing one or more non-powered carriers as a train. The AGVs are moving along a unidirectional loop guided path (Figure 9).

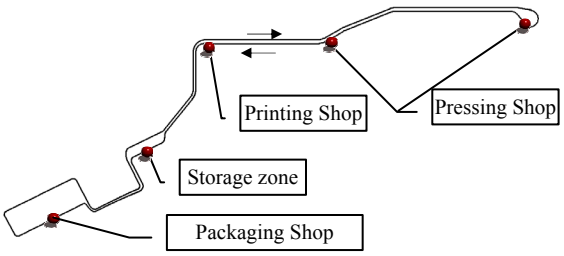


Figure 9: The transportation loop with the stations.

The simulation study is made with a couple of objectives. The first one is to determine the rule to be applied to load the carriers. We consider the two following rules:

R1: the products resulting from different production orders can be mixed on the same carrier.

R2: one carrier holds only products of the same production order.

The second objective is to locate and to size the waiting areas of the carriers. Indeed, it is necessary to place one or more garages near each workshop where the carriers can wait for a tow.
The simulator we developed shows how the 600 carriers are distributed between the different storage areas according the applied production rules. The evolution of the total number of the waiting carriers in each storage area according to time is shown on Figure 10.

6 CONCLUSION

In this paper, we presented some important problems to be considered when employing AGV systems. Some of these problems concern the design aspect, the others the piloting or control aspect. In our research work, we were concerned with the control aspect, particularly with the conflict-free routing of bi-directional AGVSs.
This paper recalls the principle of the approaches we developed for reactive routing of bi-directional AGVs and gives some simulation results. We also briefly presented a simulation study we have done to help an industrialist to make decisions at the design stage of their AGV system and also at the piloting stage. Since this application is confidential, we could not give more information on it.

REFERENCES

Albert P., 1998. Pilotage de véhicules autoguidés dans un système automatisé de production. PhD thesis – University of Nantes.

Beamon, B.M, Chen, V.C.P., 1998a. Performability-based fleet sizing in a material handling system. *International Journal of Advanced Manufacturing Technology,* Vol.14, No.6, pp.441-449.

Beamon, B.M., Deshpande, A.N., 1998b. A mathematical programming approach to simultaneous unit-load and fleet size optimisation in material handling systems design. *International Journal of Advanced Manufacturing Technology,* Vol.14, No.11, pp.858-863.

Blazewicz, J., Eiselt, H., Finke, G., Laporte, G., Wegrlarz, J., 1991. Scheduling tasks and vehicles in

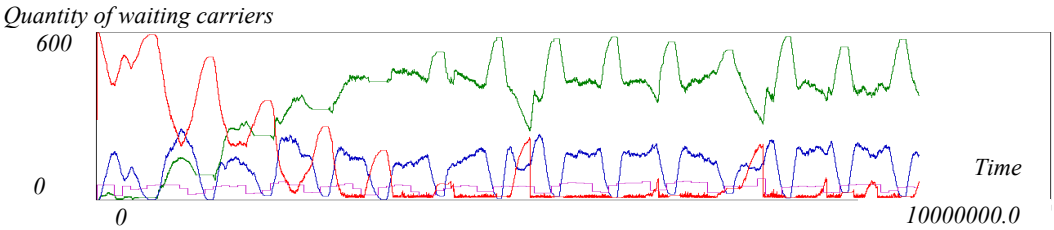


Figure 10: Example of time evolution of the carriers' number in each waiting area.

a flexible manufacturing system. *The International Journal of Flexible Manufacturing Systems,* Vol.4, pp.5-16.

Branislav, H., Mrafko, L., Bielko, V., 2002. A compraison of the AGV control approaches unsing Petri Nets. *International conference on Systems, Man and Cybernetics, Tunisia.*

Breton, L., Maza, S., Castagna, P., 2006. A multi-agent based conflict-free routing approach of bi-directional automated guided vehicles. *American Control Conference, USA*.

Castagna, P., Mebarki, N., Maza, S., 2004. Evaluation du nombre d'AGVs sur un système de production de type Job shop. *5ème Conférence Francophone de Modélisation et SIMulation (MOSIM'04),* pp.619-626, France.

Collinot, A., Droguoul, A., 1996. La méthode de conception multi-agent cassiopée: Application à la robotique collective. *Technical Report LAFORIA, LIP6*.

Egbelu, P.J., Tanchoco, J.M.A., 1984. Characterization of automatic guided vehicle Dispatching rules. *International Journal of Production Research,* Vol.22, No.3, pp.359-374.

Egbelu, P.J., 1987. The use of non-simulation approaches in estimating vehicle requirements in an automated guided vehicle based transport systems. *Material flow,* No.4, pp.209-221.

Gaskins, R.J., Tanchoco, J.M.A., 1987. Flow path design for automated guided vehicle systems. *International Journal of Production Research,* Vol.27, No.6, pp.915-926.

Krishnamurthy, N.N., Batta, R., Karwan, H., 1993. Developing conflict-free routes for automated guided vehicles. *Operations Research,* 41(6), pp.1077-1090.

Kim, C.W., Tanchoco, J.M.A., 1991. Conflict-free shortest-time bi-directional AGV routing. *International Journal of Production Research,* Vol.29, No.12, pp.2377-2391.

Lawley, M., Reveliotis, S., Ferreira, P., 1999. Design guidelines for deadlock handling strategies in flexible manufacturing systems. *International Journal of Manufacturing Systems,* Vol.9, pp.5-30.

Maza, S., Castagna, P., 2001. Conflict-free AGV routing in bi-directional networks. *8th IEEE International conference on Emerging Technologies and Factory Automation,* pp.761-764, France.

Maza, S., Castagna, P., 2005a. A performance-based structural control policy for conflict-free routing of bi-directional Automated Guided Vehicles. *Computers in industry, Elsevier publication,* Vol.56, n°7, pp.719-733.

Maza, S., Castagna, P., 2005b. Sequence-based hierarchical conflict-free routing strategy of bi-directional automated guided vehicles. *16th IFAC World Congress, Praha*.

Oboth, C., Batta, R., Karwan, M., 1999. Dynamic conflict-free routing of automated guided vehicles. *International Journal of Production Research,* Vol.37, No.9, pp.2003-2030.

Pia Fanti, M., Maione, B., Mascolo, S., Turchiano, B., 1997. Event-based feedback control for deadlock avoidance in flexible production systems. *IEEE Transactions on Robotics and Automation,* Vol.13, No.3, pp.347-363.

Reveliotis, S.A., Conflict resolution in AGV systems. *IIE Transactions,* Vol.32, No.7, pp.647-659.

Reveliotis, S.A., Ferreira, P., 1996. Deadlocks avoidance policies for automated manufacturing cells. *IEEE Transactions on Robotics and Automation,* Vol.12, No.3, pp.845-857.

Silberschatz, A., Peterson, J.L., 1991. Operating system concepts, Addison Wesley.

Taghaboni, F., Tanchoco, J. M. A., 1995. Comparison of Dynamic Routing

Techniques for Automated Guided Vehicle Systems. *International Journal of Production Research,* Vol.10, No.33, 2653-2669.

Tanchoco, J.M.A., Egbelu, P.J., 1986. Potentials for bi-directional guide-path for automated guided vehicle based systems. *International Journal of Production Research,* Vol.24, No.5, pp.1075-1097.

MODELING AND CONTROL OF AN EXPERIMENTAL SWITCHED MANUFACTURING SYSTEM

Michael Canu, Dominique Morel and Naly Rakoto-Ravalontsalama*
*Ecole des Mines de Nantes (*and IRCCyN)*
4 rue Alfred Kastler, 44307 Nantes Cedex 03, France
(mcanu,morel,rakoto)@emn.fr

Keywords: Manufacturing Systems.

Abstract: The notion of *switched discrete event systems* (DES) has been introduced recently. This is a class of DES where each automaton is the composition of two basic automata, but with different composition operators. A switching occurs when there is a change of the composition operator, but keeping the same two basic automata. A mode behavior is defined as the active DES behavior for a given composition operator. Composition operators are supposed to change more than once so that each mode is visited more than once. In this paper we study the modeling and control of an experimental manufacturing system as an example of switched DES.

1 INTRODUCTION

Supervisory control initiated by Ramadge and Wonham (Ramadge and Wonham, 1987) provides a systematic approach for the control of discrete event system (DES) plant. Most of the properties of a given composed system depend on the composition operator. The modular approach reflects the underlying physical properties of complex systems such as manufacturing systems.

The most common composition operators used in supervisory control theory are the product and the parallel composition (Cassandras and Lafortune, 1999), (Wonham, 2004). However many different types of composition operators have been defined, e.g., the prioritized synchronous composition (0), the biased synchronous composition (Lafortune and Chen, 1990), see (Wenck and Richter, 2004) for a review of most of the composition operators. Multi-Agent composition operator (Romanovski and Caines, 2002), (Romanovski and Caines, 2006) is another kind of operator, which differs from the synchronous product in the aspects of simultaneity and synchronization.

Related work concerns a) fault diagnosis for DES (the readers are referred to (Jensen, 2003) for a comprehensive survey), b) *mode-automata* for reactive system programming, introduced in (Maraninchi and Remond, 1998), c) supervisory uniqueness for operating mode systems studied in (Kamach et al., 2005) where the authors propose a multi-model approach to DES, and finally d) sensor failure tolerant supervisory control proposed in (Rohloff, 2005) where different automata are used to model the system observation behavior in the various modes of operations.

This paper studies the application of switched DES methodology to an Experimental Manufacturing Cell. This cell is composed of two robotized workstations connected to a central conveyor belt. Recently, three new semi-automated workstations have been added in order to increase the flexibility aspects of the cell. This flexibility allows the designer to study different mode behaviors of the experimental cell.

The paper is organized as follows. In Section 2, the notation and preliminaries are given. The notion of switched DES is recalled in Section 3. In Section 4, the controllability property is studied. Finally the experimental cell behavior is described in Section 5.

2 NOTATION AND PRELIMINARIES

Let the discrete event system plant be modeled by a finite state automaton (Hopcroft and Ullman, 1979)

$$G = (Q, \Sigma, \delta, q_0, Q_m)$$

where Q is the finite set of states, Σ is the finite set of events associated with the transitions in G, $\delta : Q \times \Sigma \to Q$ is the partial transition function, q_0 is the initial state and $Q_m \subseteq Q$ is the set of marked states.

Let Σ^* be the set of all finite strings of elements in Σ including the empty string ε. The function δ can be generalized to $\delta : \Sigma^* \times Q \to Q$. The notation $\delta(s,q)!$ for any $s \in \Sigma^*$ and $q \in Q$ denotes that $\delta(s,q)$ is defined. Let $L(G) \subseteq \Sigma^*$ be the language generated by G, that is,

$$L(G) = \{s \in \Sigma^* | \delta(s, q_0)!\}$$

Let $K \subseteq \Sigma^*$ be a language. The set of all prefixes of strings in K is denoted by $\overline{K}$ with $\overline{K} = \{s \in \Sigma^* | \exists\, t \in \Sigma^*; st \in K\}$. A language K is said to be *prefix closed* if $K = \overline{K}$. The event set Σ is decomposed into two subsets Σ_c and Σ_{uc} of controllable and uncontrollable events, respectively, where $\Sigma_c \cap \Sigma_{uc} = \emptyset$. A controller, called a supervisor, controls the plant by dynamically disabling some of the controllable events. A closed language $K \subseteq L(G)$ is said to be *controllable* with respect to $L(G)$ and Σ_{uc} if (Ramadge, 1987)

$$\overline{K}\Sigma_{uc} \cap L(G) \subseteq \overline{K}.$$

In the supervisory control theory, composition means synchronization of finite state automata. The basis for the definition of all the composition operators are $G_a = (Q_a, \Sigma_a, \delta_a, q_{0a}, Q_{ma})$ and $G_b = (Q_b, \Sigma_b, \delta_b, q_{0b}, Q_{mb})$ with disjoint state sets $Q_a \cap Q_b = \emptyset$ but generally overlapping event sets. The result of any composition is an automaton $G_i = G_a||_{op_i} G_b = (Q, \Sigma, \delta, q_0, Q_m)$ with the state $Q = Q_a \times Q_b$, the event set $\Sigma = \Sigma_a \cup \Sigma_b$ and initial state $x_0 = x_{0a}, x_{0b}$, where op_i is a composition operator. Each operator is defined by a distinct transition function with $\sigma \in \Sigma$ a single event and $q \in Q$ a state.

Among the different types of composition operators, we recall here the biased synchronous composition (BSC) and the strict product composition (SPC).

Definition 1 *(Lafortune and Chen, 1990)* The *Biased Synchronous Composition (BSC)* is defined as follows. The automaton G_a is called the master and G_b is called the follower.

$\delta(q,\sigma) =$

$$\begin{cases} \delta_a(q_a,\sigma) \times \delta_b(q_b,\sigma) & \text{if } \delta_a(q_a,\sigma)! \wedge \delta_b(q_b,\sigma)! \\ \delta_a(q_a,\sigma) \times \{q_b\} & \text{if } \delta_a(q_a,\sigma)! \wedge \neg\delta_b(q_b,\sigma)! \\ \emptyset & \text{otherwise.} \end{cases}$$

Definition 2 The *strict product composition (SPC)* is defined as follows.

$\delta(q,\sigma) =$

$$\begin{cases} \delta_a(q_a,\sigma) \times \delta_b(q_b,\sigma) & \text{if } \delta_a(q_a,\sigma)! \wedge \delta_b(q_b,\sigma)! \\ \emptyset & \text{otherwise.} \end{cases}$$

These two composition operators will be taken as example in the next sections.

3 SWITCHED DES

The basic idea is the following. Without loss of generality we consider two automata G_a and G_b as defined above. Let G_i be the composed automaton from G_a and G_b with operator op_i, that is $G_i = G_a||_{op_i} G_b$. In the same way let G_j be the composed automaton from the same G_a and G_b but with operator op_j, that is $G_j = G_a||_{op_j} G_b$, as it is depicted in Figure 2 and Figure 3.

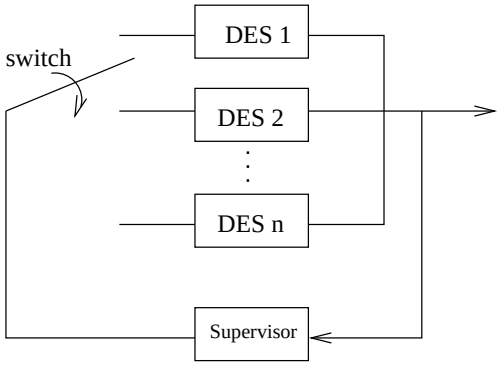


Figure 1: Switched DES.

Definition 3 Equivalent states.
The states $(q_a, q_b)^i$ of automaton G_i and $(q_c, q_d)^j$ of automaton G_j are said to be *equivalent* $(q_a, q_b)^i \equiv (q_c, q_d)^j$ if they result from the composition of the same pair of states but with different composition operators ($q_a = q_c$ and $q_b = q_d$).

Assumptions. Given two automata G_i and G_j, switching between automaton G_i and automaton G_j is possible if the following assumptions hold.

1. G_i and G_j have at least two equivalent states
2. Switching between G_i and G_j is performed through their equivalent states.
3. Switching from G_i to G_j has zero duration, as well as from G_j to G_i.

Definition 4 *(Rakoto, 2006b)* Switched DES.
A switched discrete event system is defined as follows.

$$L_{switched}(G) = L(G_i),\ i \in I = \{1,\ldots,n\} \qquad (1)$$

where G_i is the model of DES_i, and I is an index set. In this special case, $G_i = G_a||_{op_i}G_b$.

We can see in Figure 2 and Figure 3 the automaton G_1 and G_2, respectively. Then automata G_1 and G_2 can switch between them, as it is shown in Figure 4. Actually, the switching are made through the equivalent states of G_1 and G_2, see Figure 5.

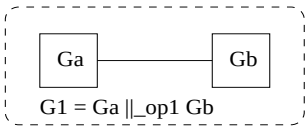


Figure 2: Automaton G_1.

Figure 3: Automaton G_2.

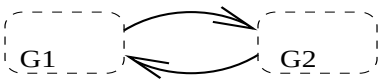


Figure 4: Switched automaton.

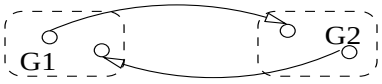


Figure 5: Automata switching through equivalent states.

We give here below some examples of switched DES:

- Manufacturing systems where the operating modes are changing (e.g. from normal mode to degenerated mode)
- Discrete event systems after an emergency signal (from normal to safety mode)
- Complex systems changing from normal mode to recovery mode (or from safety mode to normal mode).

We can distinguish, like for the switched continuous-time systems, the notion of *autonomous* switching where no external action is performed and the notion of *controlled* switching, where the switching is *forced*. The notion of switched DES has been adapted from the switched continuous-time systems. For a survey of switched continuous-time systems, one can refer to (Liberzon and Morse, 1999) and the references therein.

On one hand DES and continuous-time systems share the notion of controllability (but each domain has its own definition). On the other hand, stability analysis in continuous-time systems cannot be adapted to DES (even though some work exist on the stability of DES (Passino et al., 1994), (Passino and Burgess, 1998) and the references therein). Thus the notion of stability analysis has been changed to nonblocking analysis. Before defining the problems, we need to define the notion of switching sequence. A switching sequence is defined to be the successive active automata when the successive switchings occur. The following problems have been defined in (Rakoto, 2006b)

- *Problem A.* Find conditions that guarantee that the switched DES (1) is controllable with respect to the Language $L(G)$ and with respect to all the uncontrollable events, for any switching sequence.
- *Problem B.* Identify the classes of switching sequences for which the switched DES (1) is controllable with respect to the Language $L(G)$ and with respect to all the uncontrollable events.
- *Problem C.* Find conditions that guarantee that the switched DES (1) is nonblocking.
- *Problem D.* Identify the classes of switching sequences for which the switched DES (1) is nonblocking.

We can note that discretization of a switched continuous system (see e.g., (Rakoto, 2001) may be a solution to the adaptation to the DES context.

4 CONTROLLABILITY OF SWITCHED DES

In this section we address a specific problem related to the controllability of a switched DES (1).

Problem 1. Given a switched automaton $L_{switched} = L(G) = L(G_i), i \in I = \{1,2\}$ where $G_1 = G_a||_{op_1}G_b = G_a||_{BSC}G_b$ and $G_2 = G_a||_{op_2}G_b = G_a||_{SPC}G_b$, find the conditions that guarantee the controllability of the switched DES $L_{switched} = L(G)$.

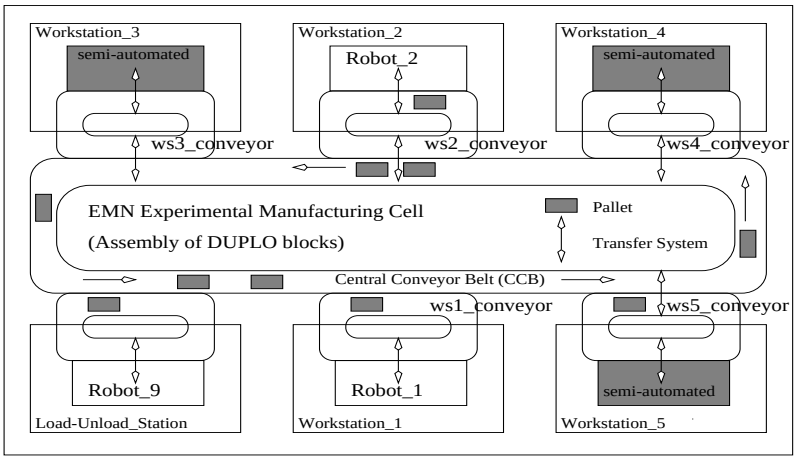


Figure 6: Experimental Manufacturing Cell.

Theorem 1 *Given a switched automaton* $L_{switched} = L(G) = L(G_i), i \in I = \{1,2\}$ *where* $G_1 = G_a||_{BSC}G_b$ *and* $G_2 = G_a||_{SPC}G_b$, *the switched automaton* $L(G) = L(G_i)$ *is controllable with respect to both* $L(G_1)$ *and* $L(G_2)$ *and with respect to* $\Sigma_u c$ *if*

1. $K_a, K_b, L(G_2)$ *are pairwise non conflicting*
2. $K_b, L(G_a)$ *are non conflicting*
3. K_b *is controllable w.r.t.* $L(G_a)$

Proof. The proof can be found in (Rakoto, 2006b). It is based on four propositions that have been given in (Wenck and Richter, 2004).

5 EXPERIMENTAL MANUFACTURING CELL

An automated manufacturing system generally consists of a number of interconnected material processing stations capable of processing a wide variety of part types, a material transport system, a communication system for integrating all aspects of manufacturing and a supervisory control system. The experimental manufacturing cell is composed of the following components (Chen et al., 2004): a) a central conveyor belt, b) two robotized workstations, with a station conveyor each, c) a transfer system between the central conveyor belt and the station conveyor, d) another transfer system between the station conveyor and the corresponding robot, and e) a load-unload robotized worksation.

Recently, three semi-automated workstations have been added to increase the flexibility aspects of the cell. Indeed, each semi-automated workstation can perform either manual or automated tasks. The experimental manufacturing cell is depicted in Figure 6.

Behavioral specifications of such an automated manufacturing system include: a) logic-based specifications (e.g. safety, error recovery, the sequencing of operations, part routing and production volume requirement), b) temporal production specification: production times, and c) utility optimality specification: e.g. costs.

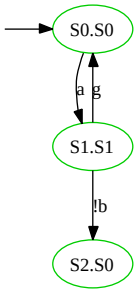


Figure 7: Composed Automaton.

The results were obtained using the tool Supremica (Akesson et al, 2006). Only two types of composition product were used. However this can be extended to different types of composition products.

6 CONCLUSIONS

This paper studies the application of the switched DES methodology, introduced previously to an Experimental Manufacturing Cell. The different mode behaviors were possible to obtain thanks to the re-

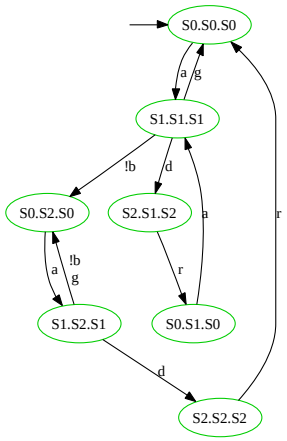


Figure 8: Composed Automaton (with priority).

cently added semi-automated workstations. These latter increased the flexibility of the system, and it allows the designer to apply the switched DES approach. Future work will be focused on a) obtaining more different mode behaviors for controllability, and b) study nonblocking properties in some specific cases.

ACKNOWLEDGEMENTS

This work was supported in part by the French 2000-2006 "Contrat Etat-Région CER STIC 9, N. 18036: Optimisation des processus industriels" Nantes, France.

REFERENCES

K. Akesson, M. Fabian, H. Flordal, and R. Malik. "Supremica - An integrated environment for verification, synthesis and simulation of discrete event systems," Proc. of 2006 Workshop of Discrete Event Systems (WODES'06), July 2006, Ann Arbor, Michigan, USA.

C.G. Cassandras and S. Lafortune. *Introduction to Discrete Event Systems*. Boston, MA, USA: Kluwer Academic Publishers, 1999.

X. Chen, D. Morel, and N. Rakoto-Ravalontsalama "Multi-Agent Based Supervisory Control of an Experimental Manufacturing Cell". Proc. of IFAC Symposium on Large Scale Systems (LSS 2004) Osaka, Japan, July 26-28, 2004, pp. 391–394.

M. Heymann. "Concurrency and discrete event control," In *IEEE Control Systems Magazine*, Vol. 10 N.4, pp. 103–112, 1990.

R.M. Jensen. "DES Controller Synthesis and Fault Tolerant Control: A Survey of recent advances," Technical report TR-2003-40, Th IT University of Copenhagen, Denmark, 2003.

J.E. Hopcroft and J.D. Ullman, "Introduction to Automata Theory, Languages, and Computation," Addison-Wesley, Reading, MA, USA, 1979.

O. Kamach, L. Pietrac, and E. Niel. "Supervisory uniqueness for operating mode systems," *Proc. of World IFAC Congress 2005*, Prague, Czech Republic, July 2005.

S. Lafortune and E. Chen. "The infimal closed controllable superlanguage and its application in supervisory control," In *IEEE Trans. on Automatic Control*, vol.35, N.4 pp. 398–405, 1990.

D. Liberzon and A.S. Morse. "Basic problems in stability and design of switched systems," In *IEEE Control Systems Magazine*, pp. 59–70, Oct. 1999.

F. Maraninchi and Y. Remond. "Mode-Automata: About modes and states for reactive systems" *Proc. of European Symp. on Programming 1998*, Lisbon, Portugal, 1998.

K.M. Passino, A.N. Michel, and P.J. Antsaklis. "Lyapunov stability of a class of discrete event systems," In *IEEE Trans. on Automatic Control*, vol.39, N.2 pp. 269–279, Feb 1994.

K.M. Passino and K.L. Burgess. "Stability Analysis of Discrete Event Systems," Wiley, New York, USA, 1998.

N. Rakoto-Ravalontsalama. "Discrete approximation of continuous and hybrid systems: Some consequences to supervisory control," *Proc. of IFAC SSC 2001 (Symp. on System Structure and Control)*, Paper N. 123, Prague, Czech Republic, Aug. 2001.

N. Rakoto-Ravalontsalama. "Supervisory control of switched discrete event systems," Proc. of Int. Symposium on MTNS 2006, July 24-28, 2006, Kyoto, Japan.

P.J. Ramadge and W.M. Wonham. "Supervisory control of a class of discrete-event processes," In *SIAM J. on Control and Optimization*, vol.25 N. 1, pp. 206–230, 1987.

K. Rohloff. "Sensor failure tolerant supervisory control" *Proc. of the 44th IEEE Conf. on Decision and Control and the European Control Conf. 2005*, Seville, Spain, pp. 3493–3498, December 2005.

I. Romanovski and P.E. Caines. "On the supervisory control of multi-agent product systems" *Proc. of the 41st IEEE Conf. on Decision and Control*, Las Vegas, Nevada USA, pp. 1181–1186, December 2002.

I. Romanovski and P.E. Caines. "Multi-Agent product systems: Controllability and Non-blocking properties" Accepted at *WODES 2006*, Ann Arbor, MI, USA, July 2006.

F. Wenck and J.H. Richter. "A composition oriented perspective on controllability of large DES," *Proc. of WODES 2004*, Reims, France, pp. 271–276, 2004.

W.M. Wonham. *Notes on Discrete Event Systems*, Dept. of ECE, University of Toronto, Toronto, Canada, 2004.

CYCLE TIME OF P-TIME EVENT GRAPHS

Ph. Declerck, Ab. Guezzi and J-L. Boimond
LISA / ISTIA, University of Angers, 62 avenue Notre Dame du lac, F-49000 Angers, France
philippe.declerck@istia.univ-angers.fr, boimond@istia.univ-angers.fr

Keywords: P-time Petri net, timed event graph, (max,+) algebra, cycle time, production rate.

Abstract: The dater equalities constitutes an appropriate tool which allows a linear description of Timed Event Graphs in the field of (max, +) algebra. This paper proposes an equivalent model in the usual algebra which can describe Timed and P-time Event Graphs. Considering 1-periodic behavior, the application of a variant of Farkas' lemma allows the determination of upper and lower bounds of the production rate and necessary conditions of consistency.

1 INTRODUCTION

Event Graphs are a subclass of Petri nets which can be used to model discrete event dynamic systems subject to saturation and synchronization phenomena, typically, transportation networks, multiprocessor systems and manufacturing systems. P-time Event Graphs are convenient tools to model systems whose operation times are included between a minimum and a maximum duration. Therefore, P-time Event Graphs can function at a maximal or a minimal speed and, average cycle time is one of the most important criteria which characterizes the system. An important result about Timed Event Graphs is that a Timed Event Graph reaches a periodic regime after a transient period (G. Cohen and Viot, 1983) (Chrétienne, 1985) in the earliest functioning mode (*i.e.*, transitions fire as soon as they are enabled). In this case, the trajectory is said K-periodic. More precisely, if $x(k)$ represents the date of firings of the transition x at the number of event k, then there is a constant λ (called the cycle time which is the inverse of the periodic throughput) and two integers k_0 in $\mathbb{N}$ and c in $\mathbb{N}^*$(called the cyclicity) such that

$$x(k+c) = x(k) + c \times \lambda \; for \; k \geq k_0$$

and

$$\lambda = \lim_{k \to \infty} \frac{x(k)}{k}$$

(Gaubert, 1995).

However, the periodical behavior is reached only after a transient that can be extremely long, moreover presence of perturbations (faults, maintenance operations,...) can limit the possibility of reaching a periodical behavior. The representativeness of the production rate can be reduced as the effectiveness of the approaches as resources optimization or control using transfert functions.

A possible approach is to generate periodic behaviors without transient period as 1-periodic behavior which is defined by

$$x(k+1) = x(k) + \lambda.$$

This technique assumes that each transition is structurally controllable (F. Baccelli, 1992).

Considering an 1-periodic behavior, the objective of the paper is the calculation of the average cycle time of P-time Event Graphs. The proposed approach introduces a new model based on "daters" in the Section 2. Defined by an inequality, the model completely describes in the usual algebra the trajectories of different Event Graphs as Timed Event Graphs or P-time Event Graphs.

Using a well-known Farkas'lemma of the linear programming (Schijver, 1987), the Sections 3 and 4 presents results about cycle time. Two examples are given in the Section 5 to illustrate the proposed method.

2 MODEL

Definition 1 *A **Petri net** is a pair (G, M_0), where $G = (R, V)$ is a bipartite graph with a finite number of nodes (the set V) which are partitioned into the disjoint sets of places P and transitions T; R consists of pairs of the form (p_i,q_i) and (q_i,p_i) with $p_i \in P$ and $q_i \in T$. The initial marking M_0 is a vector of dimension $\mid P \mid$ whose elements denote the number of initial tokens in the respective places.*

Definition 2 *For a Petri Net with $\mid P \mid$ places and $\mid T \mid$ transitions, the **incidence matrix** $W = [W_{ij}]$ is an $\mid P \mid \times \mid T \mid$ matrix of integers and its typical entry is given by $W_{ij} = W_{ij}^{+} - W_{ij}^{-}$ where W_{ij}^{+} is the weight of the arc from transition j to its output place i and W_{ij}^{-} is the weight of the arc to transition j from its input place i.*

In a Petri net, from a marking M, a firing sequence implies a string of successive markings. The characteristic vector s of a firing sequence S is a vector for which each component is an integer corresponding to the number of firings of the corresponding transition. Then a marking M reached from M_0 by firing of a sequence S can be deduced using the fundamental relation:

$$M = M_0 + W \times s$$

where M_0 is the initial marking and W is the incidence matrix.

Definition 3 *A Petri net is called an **Event Graph** if each place has exactly one upstream and one downstream transition.*

P-time Petri nets allow the modeling of discrete event dynamic systems with sojourn time constraints of the tokens inside the places. Consistently with the dioid $\overline{\mathbb{R}}_{max}$ (see ((F. Baccelli, 1992))), we associate a temporal interval defined in $\mathbb{R}^{+} \times (\mathbb{R}^{+} \cup \{+\infty\})$ for each place.

Definition 4 *A **P-time Event Graph** is a pair $< R, IS >$ where R is an Event Graph and the mapping IS: from P to $\mathbb{R}^{+} \times (\mathbb{R}^{+} \cup \{+\infty\})$ is defined by $p_i \rightarrow [a_i, b_i]$ with $0 \leq a_i \leq b_i$.*

The interval $[a_i, b_i]$ is the static interval of duration time of a token in the place p_i belonging to the set of places P. The token must stay in the place p_i during the minimum residence duration a_i. Before this duration, the token is in a state of unavailability to fire the transition t_j. The value b_i is a maximum residence duration after which the token must leave the place p_i (and can contribute to the enabling of the downstream transitions). If not, the system falls into a token-dead state. So, the token is available to fire the transition t_j in the time interval $[a_i, b_i]$.

2.1 Preliminary Inequalities

For Event Graphs, let us express the firing interval for each transition of the system guaranteing the absence of token-dead states. The set ${}^{\bullet}p$ is the set of input transitions of p and $p^{\bullet}$ is the set of output transitions of p. The set ${}^{\bullet}t_i$ (respectively, $t_i^{\bullet}$) is the set of the input (respectively, output) places of the transition t_i. The set of upstream (respectively, downstream) transitions of t_i is denoted $t_i^{\leftarrow} = {}^{\bullet}({}^{\bullet}t_i)$ (respectively, $t_i^{\rightarrow} = (t_i^{\bullet})^{\bullet}$). The following assumption alleviates the notations. We suppose that for each pair of transitions (i, j), there is at the most a unique place denoted p_{ij} between the upstream transition $t_j \in {}^{\bullet}p$ and the downstream transition $t_i \in p^{\bullet}$. Each place p_{ij} is associated with an interval $[a_{ij}, b_{ij}]$, where a_{ij} is the lower bound and b_{ij} the upper bound .

We consider the "dater" type well-known in the (max, +) algebra: each variable $x_i(k)$ represents the date of the k^{th} firing of transition x_i. If we assume a FIFO functioning of the places which guarantees that the tokens do not overtake one another, a correct numbering of the events can be carried out. In this paper, we do not take the assumption of earliest (respectively, latest) functioning which will be the subject of other studies.

Therefore, the evolution can be described by the following inequalities expressing relations between the firing dates of transitions. An Event Graph can be considered as a set of subgraphs made up of a place p_{ij} linking the upstream transition j and the downstream transition i. We denote m_{ij} the corresponding initial marking or initial number of tokens.

For the lower bounds a_{ij} of the upstream place of transition i, we can write:

$\forall x_j \in^{\leftarrow} x_i,\ a_{ij} + x_j(k - m_{ij}) \leq x_i(k)$,

or equivalently,

$x_j(k - m_{ij}) - x_i(k) \leq -a_{ij}$.

The weight 1 of $x_j(k - m_{ij})$ (respectively, -1 of $x_i(k)$) is the weight of the entering arc of the place p_{ij}, from t_j to place p_{ij} (respectively, the outgoing arc of the place p_{ij}, from place p_{ij} to transition t_i) which is equal to W_{lj}^{+} (respectively, $-W_{lj}^{-}$) if $p_l = p_{ij}$.

Respectively, for the upper bounds b_{ij} of the upstream place of transition i, we have:

$\forall x_j \in^{\leftarrow} x_i,\ x_i(k) \leq b_{ij} + x_j(k - m_{ij})$,

or equivalently,

$x_i(k) - x_j(k - m_{ij}) \leq b_{ij}$.

The weight 1 of $x_i(k)$ (respectively, -1 of $x_j(k - m_{ij})$) is the weight of the entering arc of the place p_{ij}, from t_j to place p_{ij} (respectively, the outgoing arc of the place p_{ij}, from place p_{ij} to transition t_i) which is equal to W_{lj}^{+} (respectively, $-W_{li}^{-}$) if $p_l = p_{ij}$.

2.2 Matrix Expression

Let m be the maximum number of initial tokens, the set of the previous inequalities can be expressed as follows:

$H = [H_m H_{m-1} H_{m-2} \ldots\ldots\ldots\ H_1 H_0] \times$

$$\begin{pmatrix} x(k-m) \\ x(k-m+1) \\ .... \\ x(k-1) \\ x(k) \end{pmatrix} \leq \begin{pmatrix} -A \\ B \end{pmatrix}. \qquad (1)$$

The matrix H contains the weights of the arcs entering and outgoing of the places defined above. Each place p_l linking the upstream transition j and the downstream transition i corresponds to two rows of H and particularly, $-A$ and B are vector of temporizations where $A_l = a_{ij}$ and $B_l = b_{ij}$.

Now, we consider the matrix representation in different cases: the initial marking of all places is equal to zero; the initial marking of all places is equal to one; the general case. The two last cases will be considered in the following sections.

a) The initial marking of all places is null

The evolution can be described by the following inequalities expressing relations between the firing dates of transitions:

$$\begin{cases} x_j(k) - x_i(k) \leq -a_{ij} \\ -x_j(k) + x_i(k) \leq b_{ij} \end{cases}.$$

As $x(k)$ corresponds to firing sequence S, we can deduce from the above description on the weight of the arcs that there is a direct correspondance with the incidence matrix W. Therefore, one can write the system as follows:

$$H_0 \times x(k) \leq \begin{pmatrix} -A \\ B \end{pmatrix} \qquad (2)$$

where $H_0 = \begin{pmatrix} W \\ -W \end{pmatrix}$ and $W = W^+ - W^-$.

b) The initial marking of all places is equal to one

In this case, each place initially contains only one token. One can write:

$$\begin{cases} x_j(k-1) - x_i(k) \leq -a_{ij} \\ -x_j(k-1) + x_i(k) \leq b_{ij} \end{cases}.$$

As $x(k-1)$ and $x(k)$ respectively corresponds to firing sequence S, we can deduce from the above description on the weight of the arcs that respectively, there is a direct correspondance with the incidence matrices W^+ and $-W^-$. Therefore, one can write the system as follows:

$$\begin{pmatrix} H_1 & H_0 \end{pmatrix} \times \begin{pmatrix} x(k-1) \\ x(k) \end{pmatrix} \leq \begin{pmatrix} -A \\ B \end{pmatrix}$$

with $H_1 = \begin{pmatrix} W^+ \\ -W^+ \end{pmatrix}$ and $H_0 = \begin{pmatrix} -W^- \\ W^- \end{pmatrix}$.

c) General case

Now let us give the explicit form of the system (1) or in other words, the objective is to build an equivalent model such that each place of the new Event Graph contains only zero or one token. This new form will simplify the calculations of the cycle time.

As a place contains a maximum number of m tokens, the general idea is to split each place containing m tokens into m places, where each place contains only one token.

Let us introduce the variables $\alpha^{(m-j-1)}$ for $j = 0$ to $m-1$ in the inequations, we have:

$$\begin{pmatrix} x(k-m) \\ x(k-m+1) \\ .... \\ x(k-3) \\ x(k-2) \\ x(k-1) \\ x(k) \end{pmatrix} = \begin{pmatrix} \alpha^{(m-1)}(k-1) \\ \alpha^{(m-2)}(k-1) \\ .... \\ \alpha^{(2)}(k-1) \\ \alpha^{(1)}(k-1) \\ \alpha^{(0)}(k-1) \\ x(k) \end{pmatrix}$$

with

$$\begin{cases} \alpha^{(m-1)}(k) = x(k-m+1) = \alpha^{(m-2)}(k-1) \\ \alpha^{(m-2)}(k) = x(k-m+2) = \alpha^{(m-3)}(k-1) \\ \vdots \\ \alpha^{(2)}(k) = x(k-2) = \alpha^{(1)}(k-1) \\ \alpha^{(1)}(k) = x(k-1) = \alpha^{(0)}(k-1) \\ \alpha^{(0)}(k) = x(k) \end{cases}.$$

Or equivalently,

$$\begin{cases} \alpha^{(m-j-1)}(k) = x(k-m+j+1) = \alpha^{(m-j-2)}(k-1) \\ \text{for } j = 0 \text{ to } m-2 \\ \alpha^{(0)}(k) = x(k) \end{cases}.$$

The new state vector is:

$$X = (\alpha^{(m-1)}, \alpha^{(m-2)}, \alpha^{(m-3)}, ..., \alpha^{(2)}, \alpha^{(1)}, \alpha^{(0)})^t$$

and (1) becomes

$$H' \times \begin{pmatrix} X(k-1) \\ X(k) \end{pmatrix} \leq \begin{pmatrix} -A \\ B \end{pmatrix}$$

where H' contains H with the addition of null columns.

The system must be completed with $2(m-1) \times \mid T \mid$ relations in the worst case: for $j = 0$ to $m-2$,

$$\begin{cases} \alpha^{(m-j-2)}(k-1) - \alpha^{(m-j-1)}(k) \leq 0 \\ -\alpha^{(m-j-2)}(k-1) + \alpha^{(m-j-1)}(k) \leq 0 \end{cases}.$$

Therefore, one can write the system as follows:

$$\begin{pmatrix} G_1 & G_0 \end{pmatrix} \times \begin{pmatrix} X(k-1) \\ X(k) \end{pmatrix} \leq \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

with $G_1 = \begin{pmatrix} G_{11} \\ -G_{11} \end{pmatrix}$ and $G_0 = \begin{pmatrix} -G_{21} \\ G_{21} \end{pmatrix}$.

The matrix G_{11} of dimension $((m-1)\times \mid T \mid \times m)$ as G_{21}, is an subdiagonal of identity matrices immediately above the main diagonal, while the matrix G_{21} is a diagonal of identity matrices.

Finally, we can write the algebraic form:

$$\begin{pmatrix} G \\ H' \end{pmatrix} \times \begin{pmatrix} X(k-1) \\ X(k) \end{pmatrix} \leq \begin{pmatrix} 0 \\ 0 \\ -A \\ B \end{pmatrix}.$$

3 CYCLE TIME

The aim of this part is the determination of the existence of 1-periodic trajectory in P-time Event Graphs. Let us consider an Event Graph such that $m_{ij} = 0$ or 1.

$$H \times \begin{pmatrix} x(k) \\ x(k+1) \end{pmatrix} \leq \begin{pmatrix} -A \\ B \end{pmatrix} with H = \begin{pmatrix} H_{11} & H_{10} \\ H_{21} & H_{20} \end{pmatrix} \quad (3)$$

The 1-periodic behavior can be defined by $x(k+1) = \lambda \times u + x(k)$ with $u = (1, 1, ..., 1)^t$ and the average cycle time λ.

The following result will be useful.

Corollary 1 *Farkas' lemma (variant) Corollary 7.1.e in (Schijver, 1987) (Hennet, 1989).*

Let A be a matrix and let b a vector. Then the system $A \times x \leq b$ of linear inequalities has a solution x, if and only if $y \times b \geq 0$ for each row vector $y \geq 0$ with $y \times A = 0$

Theorem 1 *The system (3) can follow a 1-periodic behavior for a given cycle time λ, if and only if, for each row vector $y \geq 0$ with*

$$y \times \begin{pmatrix} H_{11} + H_{10} \\ H_{21} + H_{20} \end{pmatrix} = 0, \quad (4)$$

we have:

$$\frac{y \times \begin{pmatrix} -A \\ B \end{pmatrix}}{y \times \begin{pmatrix} H_{10} \\ H_{20} \end{pmatrix} \times u} \geq \lambda \quad (5)$$

if $y \times \begin{pmatrix} H_{10} \\ H_{20} \end{pmatrix} \times u > 0$,

$$\frac{y \times \begin{pmatrix} -A \\ B \end{pmatrix}}{y \times \begin{pmatrix} H_{10} \\ H_{20} \end{pmatrix} \times u} \leq \lambda \quad (6)$$

if $y \times \begin{pmatrix} H_{10} \\ H_{20} \end{pmatrix} \times u < 0$,

$$y \times \begin{pmatrix} -A \\ B \end{pmatrix} \geq 0 \quad (7)$$

if $y \times \begin{pmatrix} H_{10} \\ H_{20} \end{pmatrix} \times u = 0$.

Proof: We have

$$\begin{pmatrix} H_{11} & H_{10} \\ H_{21} & H_{20} \end{pmatrix} \times \begin{pmatrix} x(k) \\ \lambda \times u + x(k) \end{pmatrix} \leq \begin{pmatrix} -A \\ B \end{pmatrix}$$

i.e.,

$$\begin{cases} H_{11} \times x(k) + H_{10} \times (\lambda \times u + x(k)) \leq -A \\ H_{21} \times x(k) + H_{20} \times (\lambda \times u + x(k)) \leq B \end{cases}$$

i.e.,

$$\begin{cases} (H_{11} + H_{10}) \times x(k) \leq -A - H_{10} \times (\lambda \times u) \\ (H_{21} + H_{20}) \times x(k) \leq B - H_{20} \times (\lambda \times u) \end{cases}$$

or equivalently,

$$\begin{pmatrix} H_{11} + H_{10} \\ H_{21} + H_{20} \end{pmatrix} \times x(k) \leq \begin{pmatrix} -A \\ B \end{pmatrix} - \begin{pmatrix} H_{10} \\ H_{20} \end{pmatrix} \times \lambda \times u. \quad (8)$$

From Farkas' lemma, we can deduce that the system (8) of linear inequalities has a solution x, if and only if $y \times (\begin{pmatrix} -A \\ B \end{pmatrix} - \begin{pmatrix} H_{10} \\ H_{20} \end{pmatrix} \times \lambda \times u) \geq 0$ for each row vector $y \geq 0$ with $y \times \begin{pmatrix} H_{11} + H_{10} \\ H_{21} + H_{20} \end{pmatrix} = 0$.

So, $y \times \begin{pmatrix} -A \\ B \end{pmatrix} - y \times \begin{pmatrix} H_{10} \\ H_{20} \end{pmatrix} \times (\lambda \times u) \geq 0$

$y \times \begin{pmatrix} -A \\ B \end{pmatrix} \geq y \times \begin{pmatrix} H_{10} \\ H_{20} \end{pmatrix} \times (\lambda \times u) = \lambda \times y \times \begin{pmatrix} H_{10} \\ H_{20} \end{pmatrix} \times u$.

In this relation, the product by u gives the addition of all columns of $\begin{pmatrix} H_{10} \\ H_{20} \end{pmatrix}$. From the sign of $y \times \begin{pmatrix} H_{10} \\ H_{20} \end{pmatrix} \times u$, the two cases (6)(5) and the relevant necessary and sufficient conditions of existence of x (7) for the system (8) can be deduced. ■

Let us note that the existence of a solution depends on λ in the two first relations contrary to the last one.

4 LINKS WITH OTHER RESULTS

We assume that $m_{ij} = 1$, which simplifies the presentation of the connections with notions of incidence matrix and P-semi flows. So, $H_{11} = W^+$, $H_{10} = -W^-$, $H_{21} = -H_{11}$ and $H_{20} = -H_{10}$. The previous theorem is now applied.

To summarize, for each row vector $y \geq 0$ with

$$y \times \begin{pmatrix} W \\ -W \end{pmatrix} = 0 \tag{9}$$

- if $y \times \begin{pmatrix} -W^- \\ W^- \end{pmatrix} \times u > 0$ then

$$\frac{y \times \begin{pmatrix} -A \\ B \end{pmatrix}}{y \times \begin{pmatrix} -W^- \\ W^- \end{pmatrix} \times u} \geq \lambda, \tag{10}$$

- if $y \times \begin{pmatrix} -W^- \\ W^- \end{pmatrix} \times u < 0$ then

$$\frac{y \times \begin{pmatrix} -A \\ B \end{pmatrix}}{y \times \begin{pmatrix} -W^- \\ W^- \end{pmatrix} \times u} \leq \lambda, \tag{11}$$

- if $y \times \begin{pmatrix} -W^- \\ W^- \end{pmatrix} \times u = 0$ then

$$y \times \begin{pmatrix} -A \\ B \end{pmatrix} \geq 0. \tag{12}$$

Moreover, we consider particular vectors y: The row-vector y can highlights the lower bounds of the temporizations A which correspond to a Timed Event Graph; The row-vector y can also highlight the upper bounds of the temporizations B. However, they give a rough estimate of λ which must be improved by considering the space of the orthogonal vectors y. Now, we successively consider the upper bounds B and the lower bounds A.

Upper bounds B

Let us consider a row-vector y such that the m first entries are null. It can be defined by the vector $y = (y_1, y_2)$ with $y_1 = 0$. From (9), we deduce that $y_2 \times W = 0$. So, $y_2 \times W^- \times u \geq 0$, then $\frac{y_2 \times B}{y_2 \times W^- \times u} \geq \lambda$.

Lower bounds A

Let us consider a row-vector y such that the m last entries are null. It can be defined by the vector $y = (y_1, y_2)$ with $y_2 = 0$. From (9), we deduce that $y_1 \times W = 0$. As $W^- \geq 0$, $y_1 \times (-W^-) \times u \leq 0$, then

$$\frac{y_1 \times (-A)}{y_1 \times (-W^-) \times u} = \frac{y_1 \times A}{y_1 \times W^- \times u} \leq \lambda. \tag{13}$$

Calculation of the state

Considering any non-negative row vector y, the set of the relations defined by (11) (respectively, (10)) gives the lower bound (respectively, upper bound) of λ_1. Given an arbitrary cycle time λ_1 satisfying (11) and (10), the objective is the calculation of the date of firing of the transitions for a given k.

As $H_{11} = W^+$, $H_{10} = -W^-$, $H_{21} = -H_{11}$ and $H_{20} = -H_{10}$, from (8), $x(k)$ must satisfy

$\begin{pmatrix} W \\ -W \end{pmatrix} \times x(k) \leq \begin{pmatrix} -A \\ B \end{pmatrix} - \begin{pmatrix} -W^- \\ W^- \end{pmatrix} \times \lambda_1 \times u$.

This inequality follows the general form $A \times x \leq B$ which can be solved by the Fourier-Motzkin algorithm.

4.1 Link with Karp's Theorem

The following well-known result is based on circuits (Gaubert, 1995).

Theorem 2 *(Karp's theorem)*

In a strongly connected system, the minimal cycle time can be defined by the maximum of the ratio of the sum of the delays to the sum of tokens, for each elementary circuit C_k, i.e.,

$$minimal\ cycle\ time\ = \max_k \left(\frac{\textit{sum of delays in } C_k}{\textit{sum of tokens in } C_k}\right).$$

Let us now consider (13). Its numerator $y_1 \times A$ is a sum of durations as $y_1 > 0$ which is the total delay in C_k.

Consider the denominator of (13): $y_1 \times W^- \times u$.

As each row of W^- contains a unique entrie different from zero which can be associated with the unique token of the relevant place, the right-multiplication by u generates a column-vector $v = (1, 1, ..., 1)^t$ whose dimension is m and which is the initial marking M_0. So, the denominator $y_1 \times W^- \times u$ is equal to $y_1 \times M_0$ which is the number of tokens in C_k at M_0. Therefore, there is a correspondance between (13) and the expression of the theorem of Karp.

Strictly speaking, the Karp's theorem can be apply even if the behavior of the graph is not 1-periodic as we suppose here.

4.2 Link with (Murata, 1989)

Another result can be found in ((Murata, 1989)). If we model a Timed Petri Net which is consistent (*i.e.*, $\exists\ x > 0$, $W.x = 0$) by assigning delay d_i to each place p_i, then it can be shown that the minimal cycle time is given by:

$\max_k(\frac{y_k.D.W^+.x}{y_k.M_0})$

where y_k is the P-semi flow k and D is the diagonal matrix of d_i,$i = 1, 2, .., m$.

So, $W^+.x = v$ and $y_k.D.W^+.x = y_k.A$ which is the numerator of (13).

5 EXAMPLES

5.1 First Example

Let us consider a simple example based on two elementary strongly connected subgraphs.

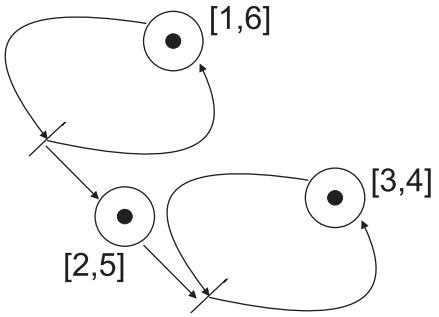


Figure 1: A simple P-time Event Graph.

$$\begin{pmatrix} W^+ & -W^- \\ -W^+ & W^- \end{pmatrix}.\begin{pmatrix} x(k-1) \\ x(k) \end{pmatrix} \leq \begin{pmatrix} -A \\ B \end{pmatrix}$$

with $x(k) = \begin{pmatrix} x_1(k) & x_2(k) & x_3(k) \end{pmatrix}^t$, $W^+ = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}$, $-W^- = \begin{pmatrix} -1 & 0 \\ 0 & -1 \\ 0 & -1 \end{pmatrix}$, $-A = \begin{pmatrix} -1 \\ -2 \\ -3 \end{pmatrix}$ and $B = \begin{pmatrix} 6 \\ 5 \\ 4 \end{pmatrix}$. We have

$$W = \begin{pmatrix} 0 & 0 \\ 1 & -1 \\ 0 & 0 \end{pmatrix}.$$

A possible integer matrix $Y \geq 0$ such that $Y.\begin{pmatrix} W \\ -W \end{pmatrix} = 0$ is as follows. $Y = \begin{pmatrix} 7 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 7 & 0 & 0 & 0 \\ 0 & 0 & 0 & 7 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 7 \\ 0 & 7 & 0 & 0 & 7 & 0 \end{pmatrix}$

$\begin{pmatrix} -W^- \\ W^- \end{pmatrix} \times u = \begin{pmatrix} -1 & -1 & -1 & 1 & 1 & 1 \end{pmatrix}^t$

$Y.\begin{pmatrix} -W^- \\ W^- \end{pmatrix} \times u = \begin{pmatrix} -7 & -7 & +7 & +7 & 0 \end{pmatrix}^t$

$Y \times \begin{pmatrix} -A \\ B \end{pmatrix} = \begin{pmatrix} -7 & -21 & +42 & +28 & +21 \end{pmatrix}^t$.

The two first terms lead to lower bounds ($\frac{-7}{-7} = 1$, $\frac{-21}{-7} = 3$), the two successive terms gives the upper bounds ($\frac{+42}{+7} = 6$, $\frac{+28}{+7} = 4$) and the last one is a condition of consistency ($+21 \geq 0$).

Therefore, the 1-periodic trajectory exists with $\max(1, 3) = 3 \leq \lambda \leq \min(6, 4) = 4$.

For $\lambda = 3$, a possible trajectory is $\binom{1}{0} \rightarrow \binom{4}{3} \rightarrow \binom{7}{6} \rightarrow ...$

For $\lambda = 3.5$, a possible trajectory is $\binom{1.5}{0} \rightarrow \binom{5}{3.5} \rightarrow \binom{8.5}{7} \rightarrow ...$

For $\lambda = 4$, a possible trajectory is $\binom{2}{0} \rightarrow \binom{6}{4} \rightarrow \binom{10}{8} \rightarrow ...$

5.2 Second Example

Now, we consider a P-time Event Graph without directed circuit.

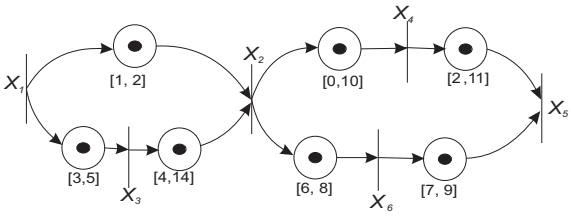


Figure 2: A P-time Event Graph.

$$\begin{pmatrix} W^+ & -W^- \\ -W^+ & W^- \end{pmatrix}.\begin{pmatrix} x(k-1) \\ x(k) \end{pmatrix} \leq \begin{pmatrix} -A \\ B \end{pmatrix}$$

with $x(k) \begin{pmatrix} x_1(k) & x_2(k) & x_3(k) & x_4(k) & x_5(k) & x_6(k) \end{pmatrix}^t$,

$W^+ = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$, $W^- = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$, $-A = \begin{pmatrix} -1 \\ -3 \\ -4 \\ 0 \\ -6 \\ -2 \\ -7 \end{pmatrix}$ and $B = \begin{pmatrix} 2 \\ 5 \\ 14 \\ 10 \\ 8 \\ 11 \\ 9 \end{pmatrix}$

$$W = \begin{pmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \end{pmatrix}$$

A possible integer matrix $Y \geq 0$ such that $Y.\begin{pmatrix} W \\ -W \end{pmatrix} = 0$ is as follows. $Y =$

$$\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0
\end{pmatrix}$$

$$\begin{pmatrix} -W^- \\ W^- \end{pmatrix} \times u =$$

$$\begin{pmatrix} -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}^t$$

$$Y.\begin{pmatrix} -W^- \\ W^- \end{pmatrix} \times u =$$

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & +1 & -1 & 0 & 0 \end{pmatrix}^t$$

$$Y \times \begin{pmatrix} -A \\ B \end{pmatrix} =$$

$$\begin{pmatrix} +1 & +2 & +10 & +10 & +2 & +9 & +2 & +18 \\ -5 & +15 & +10 \end{pmatrix}^t$$

The 9^{nd} term leads to the lower bound ($\frac{-5}{-1} = 5$), the 8^{nd} term gives the upper bound ($\frac{+18}{+1} = 18$) and the last one are conditions of consistency which are satisfied.

Therefore, the 1-periodic trajectory exists with $5 \leq \lambda \leq 18$

For $\lambda = 5$, a possible trajectory is $\begin{pmatrix} 3 & 0 & 1 & 5 & 4 & 1 \end{pmatrix}^t \rightarrow \begin{pmatrix} 8 & 5 & 6 & 10 & 9 & 6 \end{pmatrix}^t \rightarrow \begin{pmatrix} 13 & 10 & 11 & 15 & 14 & 11 \end{pmatrix}^t \rightarrow ...$

6 CONCLUSION

Using a new incidence matrix, the model we propose allows the counting of the events in Timed and P-time Event Graphs. The connections with usual incidence matrix has been realized. Considering 1-periodic behavior, the application of a variant of Farkas' lemma leads to the introduction of a generalization of the P-semi flow vectors for Timed and P-time Event Graphs, and allows the determination of upper and lower bounds of the possible cycle time. Each limit is respectively a complex function of lower and upper bounds of the temporizations. Moreover, even if cycle time λ belongs to this interval, the system must also satisfy conditions of consistency such that the finite initial dates of firing exist. With the restriction that a 1-periodic behavior has been considered, the proposed lower bound of the cycle time includes the Karp's relation.

REFERENCES

Chrétienne, P. (1985). *Analyse des régimes transitoire et asymptotique d'un graphe d'événements temporisé*. Technique et Science Informatique, pages 127-142.

F. Baccelli, G. Cohen, G. O. J.-P. Q. (1992). *Synchronization and Linearity: An Algebra for Discrete Event Systems*. Wiley.

G. Cohen, D. Dubois, J.-P. Q. and Viot, M. (1983). Analyse du comportement périodique de systèmes de production par la théorie des dioides. In *Rapport de recherche*. INRIA.

Gaubert, S. (November 1995). Resource optimization and (min, +) spectral theory. In *IEEE Transactions on Automatic Control, Vol. 40, No. 11*.

Hennet, J.-C. (1989). Une extension du lemme de farkas et son application au problème de régulation linéaire sous contraintes. In *Compte-Rendus à l'Académie des Sciences, t. 308, Série I, pp. 415-419*.

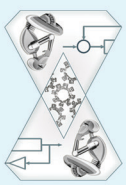
Murata, T. (1989). Petri nets: Properties, analysis and applications. In *Proceedings of the IEEE, Vol. 77, No. 4*.

Schijver, A. (1987). *Theory of linear and integer programming*. John Wiley and Sons.

AUTHOR INDEX

AUTHOR INDEX (CONT.)

AUTHOR INDEX (CONT.)



Proceedings of ICINCO 2007
Fourth International Conference on
Informatics in Control, Automation and Robotics
ISBN: 978-972-8865-82-5
http://www.icinco.org