Lab 4

Getting Started with ROS

Objective

The goals are to:
- Understand basics of ROS
- Implement simple nodes to send (publish) and receive (subscribe)
- Experience different tools in ROS to debug and visualize.

Reference Materials

- ROS Website: wiki.ros.org/ROS/Tutorials: Sections 1.5-1.6, 1.11, 1.13,
- Learning ROS for Robotics Programming by Aaron Martinez, Enrique Fernandez: Chapters 2, 3
- ROS By Example by R. Patrick Goebel: Chapter 7
- Lecture notes 22 and 23

Prelab

1. Describe what each command does:
   - $ source
   - $ catkin_init_workspace
   - $ rosrunt
   - $ rostopic
   - $ printenv | grep
   - $ catkin_make
   - $ rosnnode
   - $ rosmesg
   - $ roscore
   - $ rospack
   - $ rossrv

2. Describe what is each line for in CMakeList.txt file:
   - include_directories
   - add_executable
   - target_link_libraries

3. Describe following lines of .cpp code:
   - ros::init(argc, argv, “node_name”);
   - ros::start();
   - ROS_INFO_STREAM(“Hello, world!”);
   - ros::spinOnce();
   - ros::shutdow();
Lab Procedure

Part 1: ROS Hello-world

1. Set up ros environment. Every time you open a new terminal you need to set the environment:
   
   $ source /opt/ros/<distribution>/setup.bash

   You can add above line to the end of ~/.bashrc so that you don’t need to run it every time. After sourcing you can check by typing:
   
   $ printenv | grep ROS

2. Create a new workspace. You need to create new directory by typing
   
   $ mkdir -p ~/catkin_ws/src

   where catkin_ws is the name of your workspace directory. Initialize your workspace by typing:
   
   $ cd ~/catkin_ws/src
   
   $ catkin_init_workspace

   $ cd ~/catkin_ws

   $ catkin_make

3. Create your first ROS hello world program!
   
   $ cd ~/catkin_ws/src
   
   $ catkin_create_pkg hello_world roscpp

   Now go to src directory of the package you just built and write a hello world program into a new .cpp file (hello_world.cpp). Uncomment and edit following lines in CMakeList.txt file under the package directory.

   include_directories(${catkin_INCLUDE_DIRS})

   add_executable(hello_world_node src/hello_world_node.cpp)

   target_link_libraries(hello_world_node ${catkin_LIBRARIES})

   $ cd ~/catkin_ws

   $ catkin_make

   $ source devel/setup.bash

   every time you create new package/node or open new terminal you need to source devel/setup.bash in your workspace directory to be able to run your nodes. The catkin_make command compiles your program. Open another terminal and run

   $ roscore
This will run the ros master which will be in charge of managing nodes. Back to the original terminal and run the node you just created.

$ rosrun hello_world hello_world_node

Part 2: Creating Publisher/Subscriber Nodes
Write two programs to continuously publish and subscribe messages containing a counter. That is, the “talker” node publishes a message: “Message No. X from talker” to a topic and the subscriber node subscribes to same topic and prints the received message. You should be able to receive all messages from talker node without any missing.

**Note:** Try to understand usage of ros::spin(), ros::spinOnce(), ros::Rate and parameters of NodeHandle::advertise and NodeHandle::subscribe.

Part 3: Using ROS tools
In this part, you will experience useful tools to visualize and debug your program.

**rqt_graph:**
This tool helps you to debug your program by showing nodes configuration. You can run this tool by following command:

$ rosrun rqt_graph rqt_graph

which will show you all nodes with the published/subscribed topics and connection between nodes.

![Diagram of rqt_graph](image)

**rqt_plot:**
You can plot scalar data using this tool. To open the GUI, type following on a terminal:

$ rosrun rqt_plot rqt_plot

This will show below window:
where you can type the topic name to plot the value over time.

**ROS commands:**
ROS offers a variety of commands that can help you to debug your program. A list of some the commands are as follow:

$ rosmsg list/md5/package/packages/show
$ rosnode cleanup/info/kill/list/machine/ping
$ rosld
$ rostopic bw/echo/find/hz/info/list/pub/type

**Post Lab Questions:**
1. Try different orders of running subscriber and publisher nodes. What are you observing? Was it expected?
2. What is the difference between ros::spin() and ros::spinOnce()? Explain?
3. How did you program to receive all messages (including the first one) sent by publisher? Explain?
4. Explore rqt_graph and describe features that provides for you. Include node graph of hello_world and talker/listener in your report.
5. Explore rqt_plot and describe features that provides for you. Plot values you are getting from talker and listener and compare those numbers.
6. Explain different arguments of ROS commands and what they do?