Electrical & Computer Engineering 4330-7330
Introduction to Mechatronics and Robotic Vision
Fall/2010
Class by- DeSouza, Guilherme N. [DeSouzaG[at]missouri.edu]

Project 2

Date 12/19/2010
Prashant Revankar[prkv5[at]mail.missouri.edu]
Abstract-

There are many ways for deciding a path between two points. As a human you can decide your own path and reach the destination safe and sound but for the non-living objects to perform the same operation it takes a heavy computation and a lot of processing. The phenomenon of path planning is taken with a great interest and has been teased since decades with a considerable improvement. The improvement in the technology for the computation by the usage of GPU’s and Graphic processors make the work much faster. In proposed theory the practical problem of computation of shortest path, smoothening paths, avoidance of sharp turns, etc. is solved. These problems are of considerable importance in robotics path-planning, computational geometry, artificial intelligence, VLSI (Very Large Scale Integration), Mars (Planet) Rovers, etc. Considering set of obstacles and a string with two fixed terminal points in a 2D space, the theory will continuously tighten the string until its forms a unique configuration. The proposed theory minimizes the total length of the path creating and selecting the feature vectors in a competitive manner. The project was implemented successfully and the results with related images and videos are presented.
1. Introduction

There are various ways of path planning algorithms used by many researchers like the harmonic potential field path planning [4], the window method for the path planning [3], the rubber band technique for the planning of the path[3][4] etc. All the approaches are defining a short and smooth path. This approach gives a new way to look at the problem statement. Some use the path as a self-organizing map and homotopic path for the path planning of the objects avoiding the obstacle [1]. The proposed algorithm doesn’t consider the path to be homotopic and hence it can jump over the obstacle if needed. The algorithm is designed considering the object as a robot or a wheelchair which needs a free and smooth motion. The process is considered to be happening in the 2-D space and the obstacles and the path is preselected. The path is considered as a string which is tightened at both the ends to get the shortest possible path and also it’s smoothing. The algorithm takes care that the object passing through the path will not collide any of the obstacles and it does not depend on the number of obstacles. With giving the results of the above mentioned theory this project concludes with the demonstration of the implementation of this phenomenon.

2. Algorithm

A string ‘S’ in a 2-d space $\mathbb{R}^2$ is defined as a continuous map $S: [0, 1] \rightarrow \mathbb{R}^2$, where $S(0)$ and $S(1)$ are two terminal points of the string. The string here is not considered to be homotopic with respect to the obstacles ‘O’. Given a string $S_i$
the algorithm considering the set of obstacles computes a string $S_f$ such that the Euclidean distance covered by $S_f$ is the smallest and the smoothest.

**Fig: Flowchart for the Algorithm**
3. Objective

Assuming the string wound around the obstacles in the 2-D space $\mathbb{R}^2$ with two fixed terminal points. A shorter configuration of the strings can be obtained by pulling the strings by its terminals. The shortest configuration can be obtained by pulling its terminals until they cannot be pulled anymore. The proposed algorithm models this phenomenon of the points forming a given configuration of a string into points forming the desired shorter configuration of the string. Let us consider a set of data points or obstacles, $O = \{O_1, O_2, \ldots, O_n\}$, representing the input signals, and a sequence of variable (say, $P$) processors each of which is associated with a weight vector at any time. A weight vector represents the position of its processor in the space mentioned. If the processors are placed on a string in, the algorithm for tuning the corresponding weights to different domains of the input signals such that, on convergence, the processors will be located in such a way that they minimize a distance function, where $P_i$ and $P_{i+1}$ are two consecutive processors on the string with corresponding weights and at any time. The algorithm further guarantees that the final configuration of the string formed by the sequence of processors is the shortest path with no sharp turns.

4. Initialization

The process is initialized with a given number of connected processors, the weight corresponding to each processor on the string. A feature vector is a ‘new processor’ replace/initialized accordingly based on the situation. The weight vectors are updated iteratively on the bases of the position of the obstacle and the associated processor.
5. Creating/Choosing Feature Points

A feature vector \( X(i) \) is created if the triangular area spanned by three consecutive processors \( P_{n-1}, P_n, P_{n+1} \) does not contain any obstacle \( P \). The feature vector is created as the midpoint of the weights of \( P_{n-1} \) and the \( P_{n+1} \) at the time \( t \).

If an Obstacle lays between the triangular area spanned by the three consecutive processors \( P_{n-1}, P_n, P_{n+1} \) the feature vector is chosen to be:

\[ X_i(t) = P_j; \]

6. Analysis

The Analysis of the above mentioned algorithm is implemented, but there will be some cases in which the triangular area spanned by three consecutive processors \( P_{n-1}, P_n, P_{n+1} \) will contain more than one Obstacle ‘O’. In this case the number of obstacles is determined with their weights and a convex hull is formed as shown in figure (a) and it
overcome by adding more processors and creating a path from the processors.

7. Experimental Results

Following figures show the accuracy of the above algorithm and the implementation of it. The algorithm was tried on various different conditions as huge number of processors and very few obstacles [figure (d)], few processors and many obstacles [figure (e)], and random numbers in [figure(c)]. In [figure (f)] we can see that obstacles are in the form of walls, and the algorithm is working fine with all these possibilities.
Fig (c): Normal number of processors and obstacles

Fig (d): More processor less obstacles

Fig (e): More obstacles less processors
The Demonstration of the Iterations:

Fig (f): The Iterations of the Algorithm
8. Conclusion and Future Work

The proposed algorithm works fine with different scenarios as explained above. The algorithm is designed keeping in mind the object to be travelling on the path is a Wheelchair, so this path obtained from the algorithm is perfect for an automated Wheelchair. The proposed algorithm can also be implemented on the different mobile robots and also can be used for the surveillance applications. This algorithm is implemented in Matlab®; the future work would be implementing this algorithm in CUDA on GPU’s for the best possible performance.

9. References

[1] ”String Tightening as a Self-Organizing Phenomenon”. Bonny Banerjee
[5] Dr. G.N DeSouza, Class – 7330 Lecture-Notes