Introdcution to Robotic Systems

Final Project Report

Path planning of Autonomous Mobile robot

New Approach

Islamic University Of Gaza

Faculty of Engineering

Department of Electrical Engineering

Submitted by:

Hasan AbuMeteir — **120080503**

Walid Issa — **120060065**

Submitted to:

Assistant Professor: Dr. Iyad abuhadrous

2010-2011

Path planning algorithm

Abstract

In this present work, we present an algorithm for path planning to a target for mobile robot in unknown environment. The proposed algorithm allows a mobile robot to navigate through static obstacles, and finding the path in order to reach the target without collision. This algorithm provides the robot the possibility to move from the initial position to the final position (target). The proposed path finding strategy is designed in a grid-map form of an unknown environment with static unknown obstacles. The robot moves within the unknown environment by sensing and avoiding the obstacles coming across its way towards the target. When the mission is executed, it is necessary to plan an optimal or feasible path for itself avoiding obstructions in its way and minimizing a cost such as time, energy, and distance. The proposed path planning must make the robot able to achieve these tasks: to avoid obstacles, and to make ones way toward its target. The algorithms are implemented in Matlab, afterwards tested with Matlab GUI; whereby the environment is studied in a two dimensional coordinate system. The simulation part is an approach to the real expected result; this part is done using Matlab to recognize all objects within the environment and since it is suitable for graphic problems. Taking the segmented environment issued from Matlab development, the algorithm permit the robot to move from the initial position to the desired position following an estimated trajectory using Maps in Matlab GUI.

Introduction:

In robotic navigation, path planning is aimed at getting the optimum collision-free path between a starting and target locations. The planned path is usually decomposed into line segments between ordered sub-goals or way points. In the navigation phase, the robot follows those line segments toward the target. The navigation environment is usually represented in as configuration space. Depending on the surrounding environment and the running conditions, the optimality criterion for the path is determined. For example, in most of indoor navigation environments, the optimum path is the safest one, i.e. being as far as possible from the surrounding obstacles, whereas for outdoor navigation, the shortest path is more recommended.

The aim of this project is to compute the optimum path between a start and a target point in a given navigation map.

The idea of the project is to represent every obstacle as a charge that has a repulsive potential. In the other hand the target represent a charge has an attractive potential.

By combining these potentials a new map will be generated with an optimum path.

The algorithm:

The algorithm deals with every obstacle as a point source of repulsive potential affect on the robot with an inverse proportional of the distance square between them. This force can be computed by:

$$W/(J-y)^2+(I-x)^2$$

Where I and J represent all points in the map
X and Y represent the center of the obstacle
W represent the weight of the charge

This generate a matrix map and every element of this matrix carry the amount of potential found on (I,J) coordinates. This map will have large values at the obstacles centers and boundaries.

The other forces are generated by the target and it can be represented as a source of attractive potential and it is directly proportional with the distance with the robot. This force can be computed by:

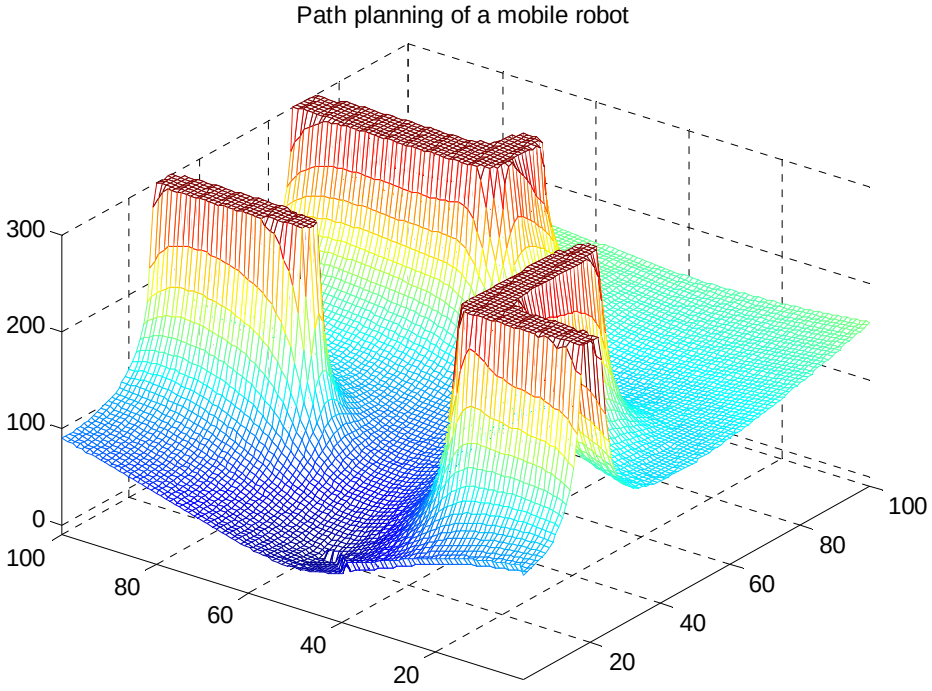
$$W*sqrt((J-GoalY)^2+(I-GoalX)^2)$$

Where GoalY, GoalX is the target coordinates

This will generate a matrix map that has a minimum value at the target and a maximum at the robot position.

Then by summing the two forces map we will get a map with repulsive and attractive forces. These forces with each other will draw the path of the robot.

Figure 1 show the two forces and as we can see the repulsive forces was drawn as a huge mount but the attractive one as a valley.



The table below is a sample of this map and show that the 300 value represent an obstacle.

131.1	141.9	155.7	174.2	199.8	237.7	297.8	300	300	300
130.9	141.8	155.8	174.5	200.4	238.6	299.6	300	300	300
130.6	141.7	155.9	174.7	201.0	240.5	300	300	300	300
130.2	141.4	155.8	174.9	201.4	241.1	300	300	300	300
129.7	141.1	155.7	175.0	201.6	240.6	300	300	300	300
129.1	140.8	155.6	175.2	202.1	241.4	300	300	300	300
128.5	140.3	155.5	175.4	202.9	243.7	300	300	300	300
127.8	139.8	155.3	175.6	203.7	245.4	300	300	300	300
126.9	139.2	155.0	175.9	204.5	246.0	300	300	300	300
125.9	138.5	154.7	176.2	205.6	248.0	300	300	300	300

Finding the path

The final part now is to find the path and this step takes the all map elements that contains the entire obstacle and goal forces values and made some steps to get the path as follows:

1. Assume the size of the map is 100*100, the algorithm first step is to begin from the first element (1,1) or origin and form a 3*3 sub map that contain the first 9 element.
2. After forming the 3*3 map from the corner of the largest map, it began to compare the 9 element with each other and determine the smallest one that represents the smallest potential force found.
3. Then make a shift of the sub map by the way that the smallest element found in step2 will be the center of the new map.
4. Comparing the smallest element coordinates with the target coordinates, if they equals then stop else continue.
5. Repeat steps 2, 3, 4 until finishing the 100*100 map.

Local minimum problem

If sometimes the robot fall in a region that results from step 2 after forming the 3*3 sub map and finding that the new target is equal to the old one and it is not equal the final coordinates, this region called a local minimum region or problem.

The proposed solution of this problem is to assume that at the local minimum point it will found an obstacle with high potential and repeating step2 will get the robot out of this region.

The example below illustrates how the algorithm works. In the first table the first element is local minimum then we replace its value with 300. After this step it found its way to the next one that also found as a local minimum then repeat until it get out.

96.9	96.91	97.1	97.5	98.2	99.2	100.6	.102	105.1	108.6
97.3	97.4	97.7	98.2	99.0	100.2	101.7	103.9	106.7	110.6
97.7	97.9	98.2	98.9	99.8	101.1	102.8	105.2	108.3	112.4
98.1	98.4	98.8	99.57	100.6	102.0	103.9	106.4	109.8	114.2
98.5	98.8	99.4	100.2	101.3	102.9	104.9	107.6	111.2	115.8
99.0	99.3	99.9	100.8	102.1	103.7	105.9	108.8	112.5	117.3
99.4	99.8	100.5	101.5	102.8	104.6	106.9	109.9	113.7	118.8
99.85	100.3	101.0	102.1	103.5	105.4	107.8	110.9	114.9	120.1
100.2	100.8	101.6	102.7	104.2	106.1	108.7	111.9	116.0	121.4
100.6	101.2	102.1	103.3	104.8	106.9	109.5	112.8	117.1	122.6

300	96.91	97.1	97.5	98.2	99.2	100.6	.102	105.1	108.6
97.3	97.4	97.7	98.2	99.0	100.2	101.7	103.9	106.7	110.6
97.7	97.9	98.2	98.9	99.8	101.1	102.8	105.2	108.3	112.4
98.1	98.4	98.8	99.57	100.6	102.0	103.9	106.4	109.8	114.2
98.5	98.8	99.4	100.2	101.3	102.9	104.9	107.6	111.2	115.8
99.0	99.3	99.9	100.8	102.1	103.7	105.9	108.8	112.5	117.3
99.4	99.8	100.5	101.5	102.8	104.6	106.9	109.9	113.7	118.8
99.85	100.3	101.0	102.1	103.5	105.4	107.8	110.9	114.9	120.1
100.2	100.8	101.6	102.7	104.2	106.1	108.7	111.9	116.0	121.4
100.6	101.2	102.1	103.3	104.8	106.9	109.5	112.8	117.1	122.6

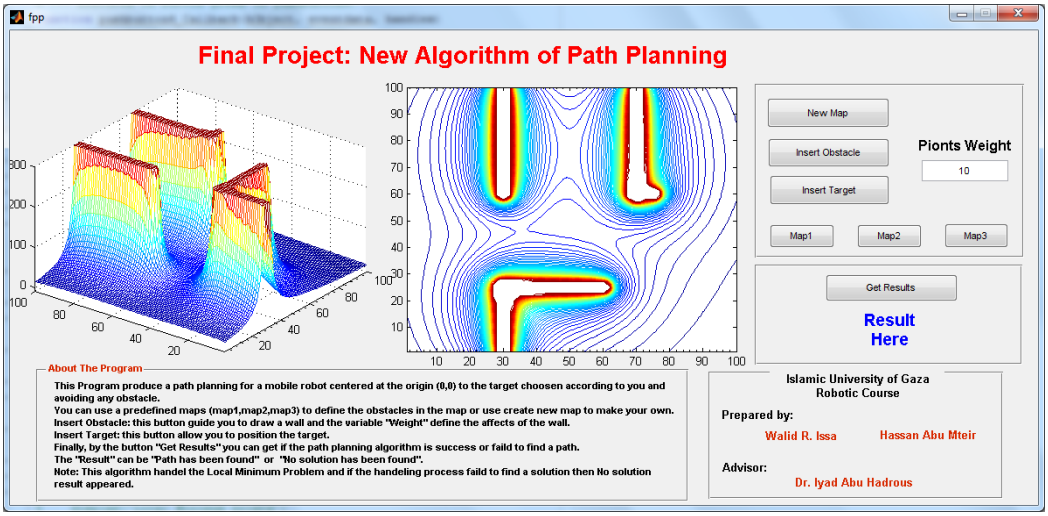
300	300	97.1	97.5	98.2	99.2	100.6	.102	105.1	108.6
97.3	97.4	97.7	98.2	99.0	100.2	101.7	103.9	106.7	110.6
97.7	97.9	98.2	98.9	99.8	101.1	102.8	105.2	108.3	112.4
98.1	98.4	98.8	99.57	100.6	102.0	103.9	106.4	109.8	114.2
98.5	98.8	99.4	100.2	101.3	102.9	104.9	107.6	111.2	115.8
99.0	99.3	99.9	100.8	102.1	103.7	105.9	108.8	112.5	117.3
99.4	99.8	100.5	101.5	102.8	104.6	106.9	109.9	113.7	118.8
99.85	100.3	101.0	102.1	103.5	105.4	107.8	110.9	114.9	120.1
100.2	100.8	101.6	102.7	104.2	106.1	108.7	111.9	116.0	121.4
100.6	101.2	102.1	103.3	104.8	106.9	109.5	112.8	117.1	122.6

300	300	300	97.5	98.2	99.2	100.6	.102	105.1	108.6
97.3	97.4	97.7	98.2	99.0	100.2	101.7	103.9	106.7	110.6
97.7	97.9	98.2	98.9	99.8	101.1	102.8	105.2	108.3	112.4
98.1	98.4	98.8	99.57	100.6	102.0	103.9	106.4	109.8	114.2
98.5	98.8	99.4	100.2	101.3	102.9	104.9	107.6	111.2	115.8
99.0	99.3	99.9	100.8	102.1	103.7	105.9	108.8	112.5	117.3
99.4	99.8	100.5	101.5	102.8	104.6	106.9	109.9	113.7	118.8
99.85	100.3	101.0	102.1	103.5	105.4	107.8	110.9	114.9	120.1
100.2	100.8	101.6	102.7	104.2	106.1	108.7	111.9	116.0	121.4
100.6	101.2	102.1	103.3	104.8	106.9	109.5	112.8	117.1	122.6

Results:

The Matlab GUI program was generated to perform this algorithm. The GUI has user friendly tools to make it easy to test all possible ideas.

Figure 2 show the interface of GUI.



The manual of using this GUI was:

This Program produce a path planning for a mobile robot centered at the origin (1,1) to the target chosen according to you and avoiding any obstacle.

You can use a predefined maps (map1,map2,map3) to define the obstacles in the map or use create new map to make your own.

Insert Obstacle: this button guide you to draw a wall and the variable "Weight" define the affects of the wall.

Insert Target: this button allow you to position the target.

Finally, by the button "Get Results" you can get if the path planning algorithm is success or failed to find a path.

The "Result" can be "Path has been found" or "No solution has been found".

Note: This algorithm handles the Local Minimum Problem and if the handling process failed to find a solution then No solution result appeared.

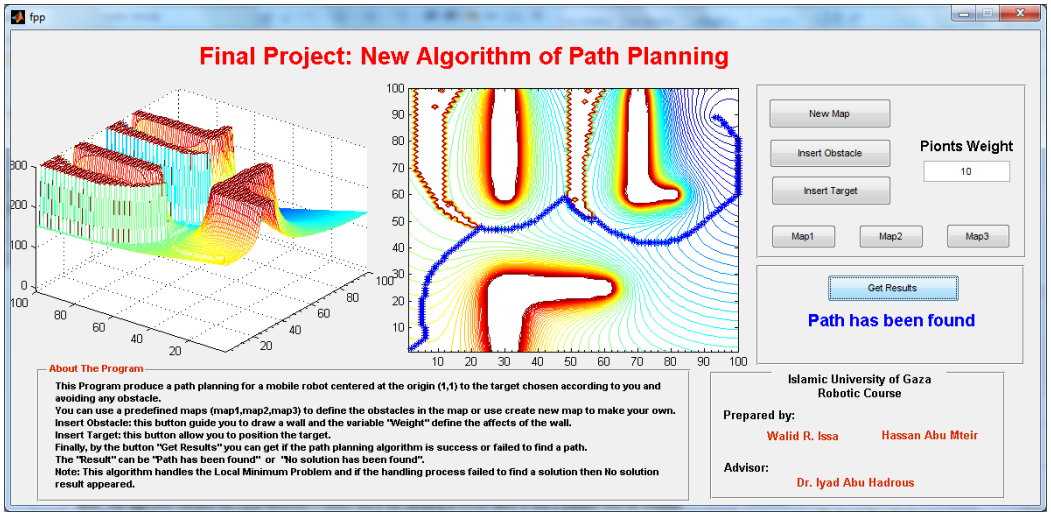
Testing the algorithm

We have 3 predefined maps to test it or we can create our own to test it

Map1:

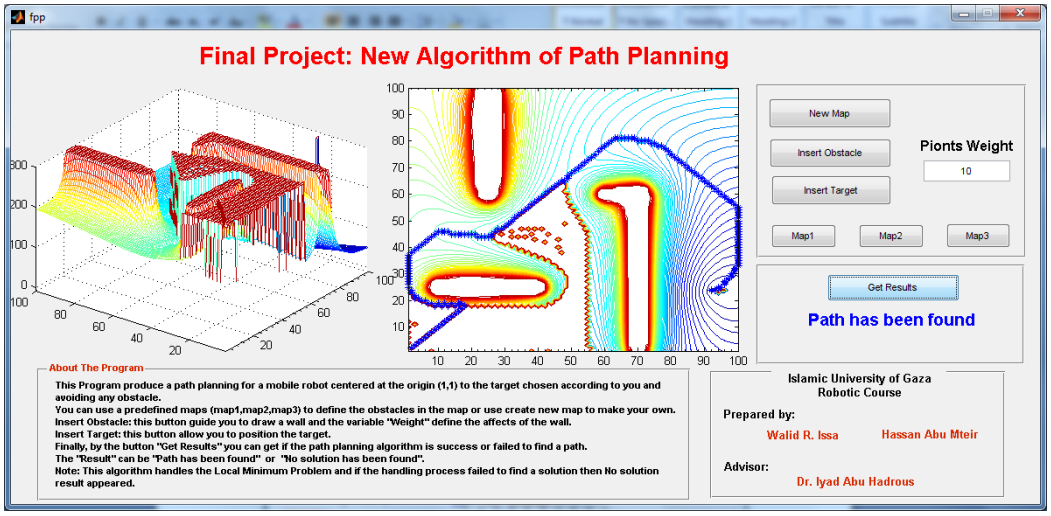
After choosing map1 and mark the target at approximately (9.4, 8.8) the right graph show the obstacles, target potential plus the path created by the algorithm. The left graph show the potential in another way.

Note that there is an extra high potential was appeared in the resulted map. This means that there was a local minimum problem was solved.



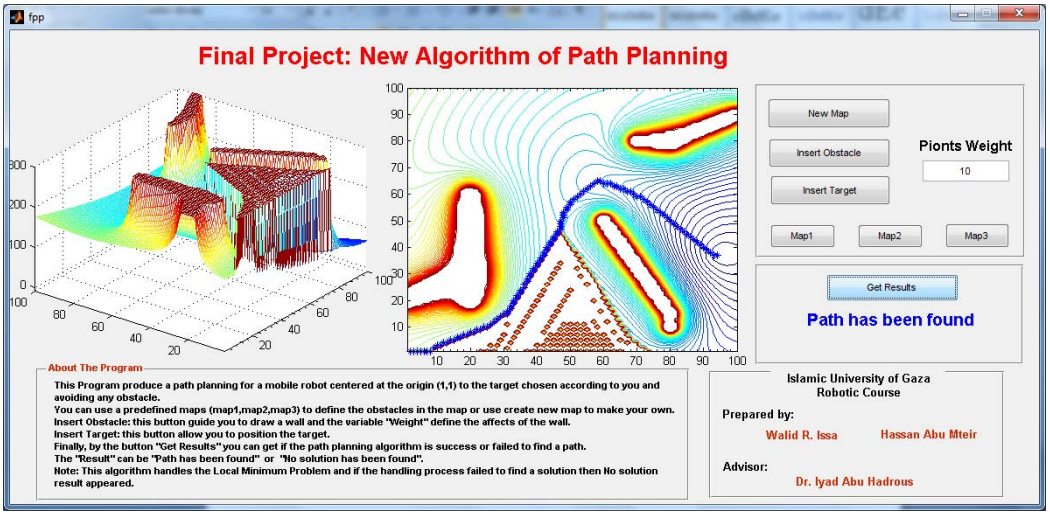
Map2:

The same was found when testing map 2



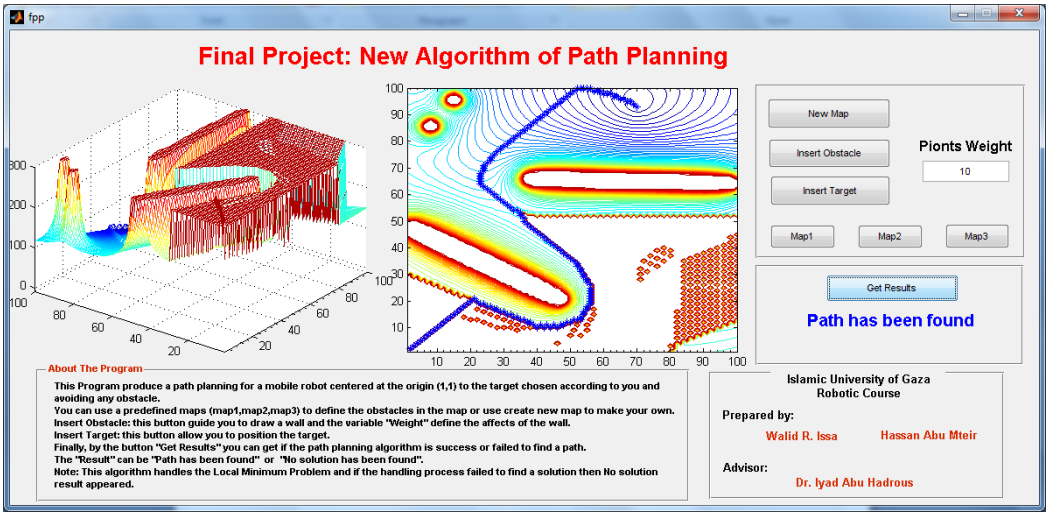
Map3:

The same was found when testing map 3



New Map:

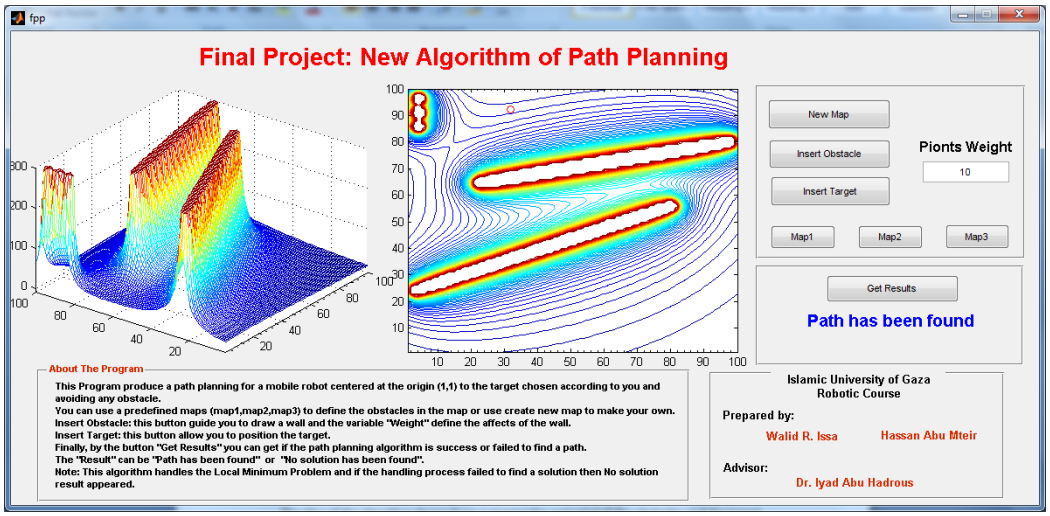
We define a map and mark a target and the result were excellent

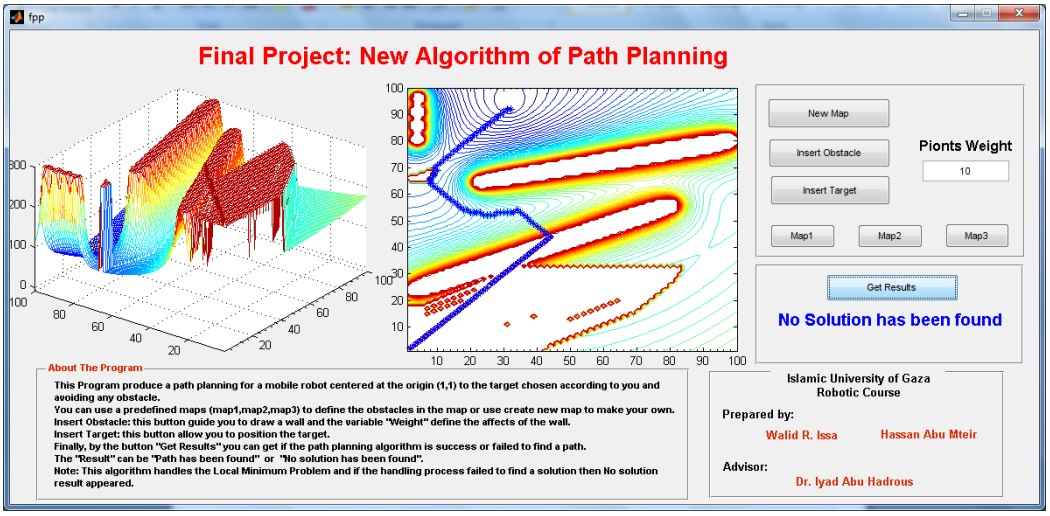


In all the previous results if we decrease the weight of the obstacles the robot will found a path easily but if increase the path will be more difficult.

New map:

The next map has obstacles as figure below and when the target has been chosen to be at (3,9.2) a "No solution" was the result and the resulting path was wrong because it cut the bottom obstacle.




fpp
Final Project: New Algorithm of Path Planning
New Map
Insert Obstacle
Insert Target
Pionts Weight
10
Map1
Map2
Map3
Get Results
No Solution has been found
About The Program
This Program produce a path planning for a mobile robot centered at the origin (1,1) to the target chosen according to you and avoiding any obstacle.
You can use a predefined maps (map1,map2,map3) to define the obstacles in the map or use create new map to make your own.
Insert Obstacle: this button guide you to draw a wall and the variable "Weight" define the affects of the wall.
Insert Target: this button allow you to position the target.
Finally, by the button "Get Results" you can get if the path planning algorithm is success or failed to find a path.
The "Result" can be "Path has been found" or "No solution has been found".
Note: This algorithm handles the Local Minimum Problem and if the handling process failed to find a solution then No solution result appeared.
Islamic University of Gaza
Robotic Course
Prepared by:
Walid R. Issa
Hassan Abu Mteir
Advisor:
Dr. Iyad Abu Hadrous

Conclusion:

In this project we propose an algorithm to find a path of a robot to the target avoiding any obstacles and solving the local minimum problems that may be faced.

The idea of the algorithm depends on summing the potential of the obstacles and the target then finding the path at the points where the potential is minimum.

The results of the algorithm were very successful.

The disadvantage of this algorithm is the robot must know its location and the target location beside the obstacles to get the path.

References:

1. www.wseas.us/journals/saed/saed-45.pdf
2. Matlab Help Center.