POTENTIAL FIELDS OBSTACLE AVOIDER

Plamena Petrova
Cornell University, Ithaca,NY
pnp4@cornell.edu

William Smart, Cindy Grimm
Department of Computer Science and Engineering
Washington University in St. Louis, St.Louis,MI.
wds/cmg@wustl.edu

Key words: Potential Fields, Obstacle-Avoidance, Linear and Circular Obstacles

Abstract.

The focus of my research was to design an obstacle avoider with SICK laser readings. The Obstacle Avoider was the beginning of a larger project. The Potential Fields algorithm was implemented, which represents the target point as an attractive force and the obstacles as repulsive forces. Then it uses those values to make a vector that repulses the robot. The two vector sums are added together to find the resultant vector along which the robot should move.

1 Introduction

Obstacle avoidance is a problem encountered by many when navigating a robot. To build a robust avoider completely aware of its surroundings is difficult especially when the algorithm one implements has drawbacks such as the Potential Fields' minimums. Altering an approach to this problem by adding in extra navigation rules enables the avoider to overcome its weaknesses and function if not perfectly, then at least satisfacty. The following two sections discuss some of the drawbacks of the Potential Fields algorithm and several tools that were implemented to counteract them.

2 Potential Fields Algorithm

Often when potential fields are used a minimum forms between two obstacles and the algorithm stops to correctly navigate the robot. Two of the most common occurences include the robot infinetely spinning around its center or turning around and heading in a direction close to that from which it came to the minimum location.

Because the force vectors depend on the inverse of the distance squared the closer the robot is to the obstacles the larger their force vectors. However, the vector sum becomes too large and offsets the general direction of the robot to the target. Thus, to counteract that, the force vector is normalized to a constant magnitude that depends on a constant radius slightly larger than the robot's radius around the obstacles.

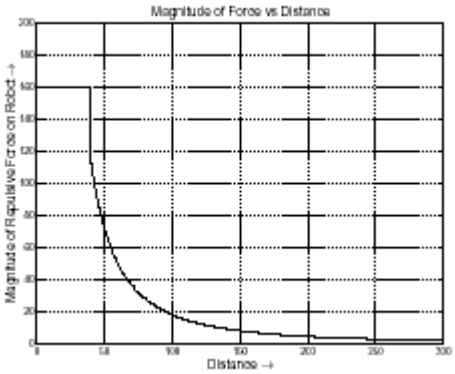


Figure 1: Variation of magnitude of force with distance for a single obstacle instance.(Mohan,3)

$$\mathbf{F_{max}} = \frac{K}{R_{min}^2}\hat{\mathbf{u}}$$

$$\mathbf{F_o} = \begin{cases} K\frac{Q\,q_o}{d^2}\hat{\mathbf{u}} & \text{if } d > R_{min} \\ \mathbf{F_{max}} & \text{if } d \le R_{min} \end{cases}$$

Therefore, the potential fields function is

piece-wise with a constant part and a parabolic part.

3 Additional Tools

This section lists several tools that were used in addition to the potential field algorithm.

3.1 Corridor

In attempt to prevent the robot from getting too close to an obstacle, I tried to implement a corridor between the obstacles along which the robot will move as described by Mohan. He proposes to build a corridor between two obstacles that the robot has to pass through by using the midpoint of the distance between the two obstacles. The corridor is parallel to the normal of the distance vector at the midpoint and is of some fixed length that is based on the radius of the robot. If the robot is within that corridor then it can proceed to move forwards parallel to it. However, if it is outside of it, a temporary goal vector is assigned to the midpoint that pulls the robot towards it. Afterwards the robot resumes to the original goal vector once inside the corridor.

To find the midpoint I decided to pick a point (point A) on the right side of the robot (laser readings from -Π/2 to 0 radians) and to calculates its x coordinate in the robot frame of reference. Then I would search through the points in the 0 to Π/2 range for a point (point B) with a close enough x coordinate to the first point. Afterwards find the distance between the two and the slope of the line. From that information i can find the normal to the distance vector from point A to point B and construct a temporary goal vector if it is needed.

However, I ran into some problems. It was hard to find correct points when the robot was facing the two obstacles from a very small angle. Also it was hard to know when the robot had two obstacles in front of it and when it didn't. Consequently, often there were several points with the same x coordinate. There were two possible coordinates: the pair of points was to be either at the same distance from one of the absolute coordinate frame axes or at the same distance from the x-axis of the x-axis of the robot frame of reference. While the latter was not hard to find when the robot is facing parallel to the normal of the distance, it was difficult to pinpoint either of the two when the robot was at an angle to the normal.

I was able detect the obstacles but finding two points to measure their midpoint was frustrating and I abandoned the approach. However, I plan to return to it later because I believe that in some cases it may optimize the trajectory.

3.2 Visible and Non-Visible Modes

The laser readings were used to designate the target as either visible or non-visible. A target that lies on a line from the robot's origin through the target without any obstacles on the same line and is stationed in the 180 degrees in front of the robot is classified as visible. Similarly, a target that does not lie on such a line is non-visible.

Therefore if the target is visible the avoider navigates the robot towards it in incremental distances as to be able to detect any new obstacles that may appear on its path and to react to the change of its environment. The increments prevent the robot from crashing with a new obstacle. On the other hand, if the path to the target is obstructed, the avoider navigates the robot though a path dependent on the surroundings.

3.3 Obstacle Region Segmention

The 180 degrees window laser readings in front of the robot were used to segment the front window into obstacle and free areas. After some testing, a close enough distance for two points to count as one segment was estimated at 10cm. This is satisfacty because it is a sixth of the robot's diameter (60cm). Due to the low

average speed of the robot, far away obstacles were excluded from the calculations. The final obstacle distance at which obstacles were classified as segments is 1m.

3.4 Obstacle Vector Weight Difference

I implemented a new function, which accounts better for the relative position of the obstacle to the robot. Before the function added all of the obstacle vectors. As a result, the vector sum wa too large or too small in some instances and thus gave a wrong vector to the vector mover. With the new function if the obstacle is to the left of the robot(0 to Π/2 radians), then the x-component of that obstacle vector would be added while the y-component of the vector would be subtracted. Similarly, if the obstacle is on the right side of the robot(0 to -Π/2 radians) the y-component of the obstacle vector would be added and the x-component subtracted.

3.5 Force Weighing Experimental Results

At the present the target force vector has a constant magnitude of 300, while the obstacle force vector depends on the inverse of the square of the distance between the obstacle and the robot.

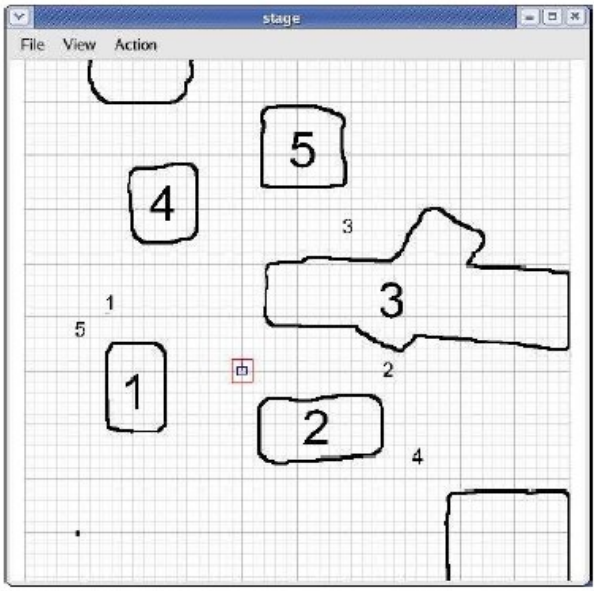


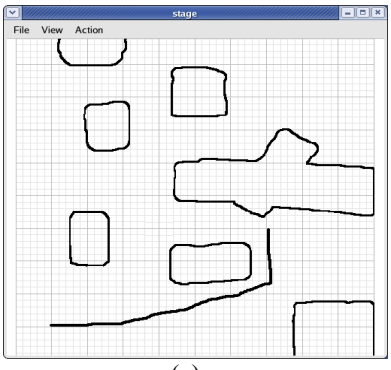
Figure 2: A snapshot of the simulator used for testing with five numbered obstacles and five target locations.

The obstacle avoider successfully navigates the robot through a sort of a tunnel composed of closely stationed objects about two meters apart and reach its target such as target point two (Fig. 2).

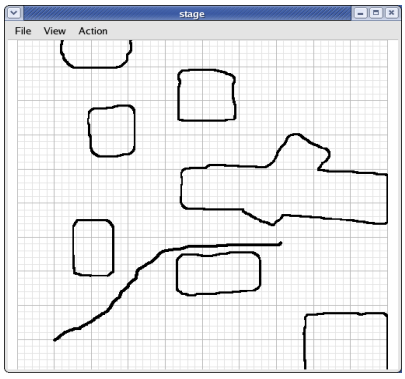
While target Five is a visible targets Two and Four are not, the avoider frequently navigated the robot in a direction in which an obstacle was present when it had finally seen its target although somewhere in front of the robot obstacles existed. Target One is not clearly seen from the bottom left corner dot but as the robot approaches the top left corner of obstacle 1 the laser can see the target. Then it would decide that it can reach the target when in fact it will crash with the obstacle. I implemented several measures hoping to correct the error, one of which includes a check for obstacles in the front 60 degrees view. However that did not work.

4 Unexpected Behavior

During the seventh week of the program, we left for AAAI (Fig.3). When we came back and began to work, most people's code did not work as it had before. After the conference, the problem described above was no longer present (crashing into an obstacle close to the target when the target is visible). However, the avoider could not navigate the robot through the areas it could before.



(a)



(b)

Figure 3: A path from (1,1) to (6,3) on which the avoider navigated the robot (a) before AAAI (b) after AAAI.

5 Conclusions and Further Work

The vector weighing can be further if the -Π/2 to Π/2 interval is divided into four or intervals, where the front two regions would influence the movement front to back while the side regions would influence the top-down movement.

A Kalman filter would lower the effect of the noise on the laser readings and would help optimizing the path to the target by eliminating paths that do not fit the predictions.

Unfortunately, due to a server problem that did not allow me to connect to the robot, my code could not be tested. One of the things i worry about if my code is tested is the difference in the laser readings. While the Player Stage simulator utilizes 360 readings in a hemisphere the actual laser which Lewis uses has only 180 degrees. That should affect the navigation since the function adds all of the readings.

6 Acknowledgements

This research has been supported by the CRA-W Distributed Mentoring Program for the 2006 summer.

Special thanks to Cindy Grimm and William Smart from the Computer Science and Engineering department at Washington University in St.Louis for their guidance and support of this project.

7 References

Mohan,M. *Integrating a Potential Field Based Pilot into a Multiagent Navigation Architecture for Autonomous Robots.*Indian Institute of Technology, Kampur, India.tu