Application of Virtual Windows to Determine the Path of a Uniformly Moving Obstacle

Hj. M.A. Hj. Mansor  
Faculty of Electrical Engineering  
MARA University of Technology  
40450 Shah Alam, Selangor  
pakngah60@gmail.com

Zarina Abdul Rashid  
Faculty of Electrical and Electronic Engineering  
University Tun Hussein Onn Malaysia  
86400 Parit Raja, Batu Pahat, Johor  
zarinar@uthm.edu.my

Erma Rahayu Mohd Faizal Abdullah  
Dept. of AI, Faculty of Computer Science & IT  
Malaya University  
50603 Kuala Lumpur  
ermarahayu81@gmail.com

Abstract – Two virtual windows are used to determine the path of a single uniformly moving obstacle. If the path of the obstacle crosses the two virtual windows, then its path can be easily determined. A few simulations were implemented to ascertain the viability and effectiveness of this technique. A table with recommended maximum speed of the uniformly moving obstacle was also provided, given the specified pixel resolution and gap between the virtual windows.

Index Terms – virtual window, uniformly moving obstacle, path determination.

I. INTRODUCTION

In recent years, many researchers have their attention pointed to the solution of obstacle avoidance in path planning. This in part is due to the extension and interest in cruise control in land vehicles, unmanned vehicle and autonomous mobile robots in an environment cluttered with obstacles. Path planning is an important problem in the navigation of autonomous mobile robots. There are many path planning algorithms with obstacle avoidance, such as potential field [1][2][3][4], visibility graphs [5][6], grid methods [7], Lee-Algorithm [8][9], and virtual window [10].

Potential field method assumes that all entities in the environment generate an artificial field around themselves in such a way that a mobile robot is attracted to its goal or target, while at the same time is repulsed by obstacles. The potential field approach can be used as a global motion planning algorithm.

The visibility graph method constructs a graph of vertices of polygons representing obstacles. It means that two vertices are connected in the graph if they are mutually visible.

Lee's algorithm is a path finding algorithm and is normally applied for the placement of circuits on the printed circuit board. It guarantees to find a path between two points if it exists.

This work concentrates on determining the path of a uniformly moving obstacle using two virtual windows. Once the path is determined, the mobile robot can then decide the next step in its action to avoid colliding with the obstacle.

II. METHODOLOGY

A. Discussion of the Virtual Window

A virtual window is basically a rectangular plane that is projected ahead of a mobile robot for the purpose of detecting obstacles. In a vision system, an image of the view forward of the mobile robot is captured. This contains information of not only the plane of interest, but also of anything before and after that plane. The image maybe sharp at the plane of interest and blur in the vicinity of the plane of interest. A virtual window captures image just of the plane of interest. Information before and after the plane is disregarded. A full and complete discussion of the virtual window and its implementation is in [11]. The intersection of this virtual window with an obstacle will provide the mobile robot with the position of the obstacle. With this information, the mobile robot can then decide on the appropriate step to avoid collision with the obstacle.

Consider a single virtual window having a size of 1m × 1m and is placed 1m away from the mobile robot. As in a digital display or camera, a display resolution can be associated with this virtual window. This resolution is very much similar in concept to the pixels in a charge-coupled device chip.

The size (length and height of the virtual window) and pixel resolution can be set to any value that is required. Ideally, the size is usually rendered a little bigger than the size of the mobile robot. This allows for a bigger forward image to be monitored and thus a greater amount of information available for processing. This will give better protection to the mobile robot from collision with a moving obstacle compared to a similar or smaller size virtual window.
Let’s assumed that the width of an obstacle in this scenario is not less than 0.1m, thus in order not to miss detecting the intersection with the virtual window, the recommended lowest resolution of the virtual window must be at least 10 × 10 pixels (Fig. 1). Assuming the speed of light in air to be $2 \times 10^8$ ms$^{-1}$, then the time taken for a single laser beam to a pixel on the virtual window and back to the sensor is about 10nsecs. For a total pixel count of 100, the total time is 1μsecs.

For smaller obstacle sizes, higher resolution is recommended, though this will impact the processing time. However, for all practical purposes, current mobile robots are much bigger than 0.1m [12][13][14].

As another example, consider an obstacle having a width of 0.01m. The recommended lowest resolution for the virtual window is then at least 100 × 100 pixels. For this total pixel count of 10,000, the total time taken for the scanning of the whole virtual window is 100μsecs or 0.1ms.

**B. Discussion of the implementation of two virtual windows**

The aim of this research work is to determine the path of the uniformly moving obstacle. For any path to be determined there must be at least two points of intersection. Since the obstacle is a uniformly moving object (having a straight path with constant speed), then obviously there must be two virtual windows placed forward of each other in order for the two intersections to occur (Fig. 2). Intersections with the two virtual windows will give sufficient information to calculate the path of the uniformly moving obstacle.

Similarly, the maximum theoretical speed for a gap of 2cm and pixel resolution of 100 × 100 is 100ms$^{-1}$. Maximum theoretical speeds for values of the other two parameters are shown in Table I.

| Gap bet. virtual windows (cm) | Pixel resolution |
|------------------------------|------------------|
| 10 × 10                      | 50ms$^{-1}$      |
| 20                          | 100ms$^{-1}$     |
| 50                          | 250ms$^{-1}$     |
| 100                         | 500ms$^{-1}$     |

**III. SIMULATION AND RESULTS**

**A. Determining the path of the obstacle with respect to the step time of the simulation**

For this simulation using Microsoft Excel™, the nearest virtual window (first virtual window) to the mobile robot is set at 1m (or 100cm), while the second virtual window is located a further 0.02m (or 2cm) from the first virtual window (Fig. 3). The start position of the obstacle is (110, 112) and the end point is (2, 4). The obstacle moves in a straight line as shown by the red dashes. The resolution of the virtual windows is 100 × 100 pixels and it is located 100cm from the mobile robot, thus the time it takes to completely scan the two virtual windows is about 0.2ms (0.0002s). The relative speed of the obstacle is assumed to be 5ms$^{-1}$ (or 500cms$^{-1}$).

![Fig. 3: Top view showing relative position of the mobile robot and the obstacle, along with other relevant parameters.](image)

It is obvious from Fig. 3 that the straight line equation of the path of the obstacle is:

$$y = x + 2$$  \hspace{1cm} (1)
This equation will be used to verify the result from the simulation. The path of the moving obstacle will be calculated using the points of intersection with the two virtual windows.

The simulation is basically the calculation of the position of the moving obstacle every 0.0002s. A sample of the calculations for the positions of the obstacle every 0.0002s is shown in Fig. 4. For this calculation, the speed of the obstacle is given as 500cms\(^{-1}\) (or 5ms\(^{-1}\), which is half of the theoretical maximum speed as shown in Table I), thus having component speeds for both \(x\) and \(y\) direction as 353.5534cms\(^{-1}\).

From Table II, for the second virtual window, the intersection is only detected when the location of the obstacle is at (99.4, 101.4). The intersection coordinates with the first virtual window is at (98, 100). The coordinate (100.1, 102.1) is not valid because the obstacle has not intersect the second virtual window yet.

Using Eqn. 1 for the path of the moving obstacle, conventional calculation gives the intersections as (100, 102) with the second virtual window, and (98, 100) with the first virtual window. The offset percentage error for the coordinates at the second virtual window is around 0.04%, while the percentage error at the first is around 0.02%.

From the coordinates of intersection in Table II, the equation of the path of the obstacle is found to be:

\[
y = x + 2
\]  

The path of the obstacle obtained through the simulation is exactly the same as the set path of the obstacle (Eqn. 1). Figure 5 shows the path of the obstacle achieved through simulation and conventional calculation. The reason for the offset is that the system is assumed to detect the intersection with the second virtual window at \(y=101.96\), instead of 102. This is because the calculation was done in steps of 0.0002s, corresponding to the time taken for the system to scan the two virtual windows. The errors as shown above are all very small, less than 0.05%. The average offset percentage error is calculated by taking the average of all the four errors at all the coordinates. This average offset percentage error is found to be 0.03%

For comparison, another example is given with a time step of 0.005s. Figure 6 show a sample of the output from the simulation and Fig. 7 the path plotted as a graph.

![Fig. 4a: A sample of the calculations as an image captured from Microsoft Excel™ at a time step of 0.0002s.](image1)

![Fig. 4b: A sample of the results with the two intersections highlighted and high precision simulated values.](image2)

![TABLE II. THE POSITION COORDINATES OF THE MOVING OBSTACLE IN THE VICINITY OF FIRST (Y=100CM) AND SECOND (Y=102CM) VIRTUAL WINDOWS.](table1)

![From the coordinates of intersection in Table II, the equation of the path of the obstacle is found to be:](equation2)
The simulated path is the same as the calculated path. Again, the offset percentage error for the coordinates at the second virtual window is around 0.6%, while the percentage error at the first is around 0.4%. The average offset percentage error is found to be 0.49%.

B. Determining the path of the obstacle with respect to the speed of the moving obstacle

The previous exercise showed the simulation with different time steps. In this exercise, a different speed for the moving obstacle will be implemented and the resulting data will be discussed.

This indicates that the technique with the two virtual windows is still able to determine the path of the moving obstacle even though the offset percentage error is large, so long as the two intersections with the two virtual windows are detectable. What this means is that even though the path is obtainable, the system (the software and firmware) may not be able to indicate with a high accuracy the value for the intersection if it is too far away from the virtual windows. Of course, from the two examples above (Fig. 5 and Fig. 7), the offset percentage error is less than 1%. Figure 8 depicts how this discrepancy can happen. The laser system was not able to detect the exact intersection coordinates, thus, the actual point of measurement is different from the point of intersection.

If only one intersection can be detected, then obviously there is no way to find the path of the moving obstacle. This is shown in Fig. 9 where the time step is at 0.007s.
From Table III, for the second virtual window, the intersection is only detected when the location of the obstacle is at (99.8, 101.8). The intersection coordinates with the first virtual window is (96.4, 98.4).

| Virtual window | Coordinates (cm) |
|----------------|------------------|
| Second         | 99.82            |
|                | 101.82           |
| First          | 96.42            |
|                | 98.42            |

The simulated path has the same slope or gradient and the same y-intercept as the calculated path (Fig. 10). However, the offset percentage error is relatively large at 1.6% for the first virtual window. The offset percentage error with the second virtual window is 0.2%. The average offset percentage error is 0.89%.

![Fig. 10: Comparison of the simulated path against the calculated path of the moving obstacle with speed of 120ms⁻¹.](image)

**IV. CONCLUSION**

It was shown that the system was able to determine the path of the moving obstacle so long as there is an intersection each with the two virtual windows. The best average offset percentage was found to be 0.03%. The other two exercises gave average offset percentage error of more than 0.1%. These are to be expected since one of the exercise a time step of 0.005sec. The time step is too large to make a fine detection. The third exercise has the speed of the obstacle at 120ms⁻¹, which is larger than the suggested maximum speed for the resolution and gap size.

This bodes very well for the system as the offset error does not affect the correct determination of the path of the uniformly moving obstacle.

**ACKNOWLEDGMENT**

The authors would like to acknowledge that this research project is funded from a grant awarded by the Research Management Institute of Universiti Teknologi MARA under the Research Intensive Faculty scheme: 600/RMI/DANA 5/3/RIF (190/2012).

**REFERENCES**

[1] Ge SS & Cui YJ, “Dynamic motion planning for mobile robots using potential field method”, Proc. IEEE Int. Conf. Autonomous Robots, vol. 13(3), pp. 207 – 222, 2002

[2] Yin L et al, “A new potential field method for mobile robot path planning in the dynamic environments”, Asian Journal of Control, vol. 11(2), pp. 214 – 225, 2009

[3] Sugiyama S et al, “Path planning of a mobile robot for avoiding moving obstacles with improved velocity control by using the hydrodynamic potential”, IEEE Int. Conf. on Intelligent Robots and Systems, pp. 1421-1426, 2010

[4] Liu C et al, “Path planning of mobile robot using new potential field method in dynamic environment”, IEEE Int. Conf. on Natural Computation, pp. 1011 – 1014, 2011

[5] Li L et al, “Present state and future development of mobile robot technology research”, Robot, vol. 24(5), pp. 475 – 480, 2002

[6] Raja P & Pugazhenti S, “Path planning for a mobile robot in dynamic environments”, Int. Journal of Physical Sciences, vol. 6(20), pp. 4721 – 4731, 2011

[7] Boschian V & Pruski A, “Grid modeling of robot cells: A memory-efficient approach”, Journal of Intelligent and Robotic Systems, vol. 8(2), pp. 201 – 203,1993

[8] Maciej P et al, “Lee-algorithm based path replanner for dynamic environments”, Int. Conf. on Signals and Electronic Systems, 2012

[9] Maciej P et al, “Obstacle avoidance procedure and Lee algorithm based path replanner for autonomous mobile platforms”, Int. Journal of Electronics and Telecommunications, vol. 59(1), pp. 85 – 91, 2013

[10] Mansor MA & Morris AS, “Path planning in unknown environment with obstacles using virtual window”, Journal of Intelligent and Robotic Systems, vol. 24(3), pp. 235 – 251, 1999

[11] Mansor MA, “Modelling and Simulation of Mobile Robot Manipulators Moving in a Space with Obstacle”, PhD Thesis, University of Sheffield, May 1998

[12] http://www.robotshop.com/4wd-omni-directional-mobile-robot-kit-2.html

[13] http://www.dfrobot.com/index.php?route=product/product&product_id=361

[14] http://www.coreconagvs.com/products/

[15] Chung W et al, “High speed navigation of a mobile robot based on robot’s experiences”, Proc. of the JSME Annual Conference on Robotics and Mechatronics, 2006