Indonesian Journal of Electrical Engineering and Computer Science
Vol. 13, No. 1, January 2019, pp. 94~101
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v13.i1.pp94-101

Application of virtual windows to determine the path of a uniformly moving obstacle

M.U. Kamaluddin, Hj. M.A. Hj. Mansor
Faculty of Electrical Engineering, MARA University of Technology, 40450 Shah Alam, Selangor, Malaysia

Article Info

Article history:

Received Aug 14, 2018
Revised Oct 15, 2018
Accepted Oct 29, 2018

Keywords:

Path determination
Uniformly moving obstacle
Virtual window

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 simulation is implemented to ascertain the viability and accuracy of this technique.

Corresponding Author:

M.U. Kamaluddin,
Faculty of Electrical Engineering,
MARA University of Technology,
40450 Shah Alam, Selangor, Malaysia.
Email: mohdu833@salam.uitm.edu.my

1. 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 [1-4], visibility graphs [5-6], grid methods [7], Lee-Algorithm [8-9], and virtual window [10].

Autonomous systems [11] allow a vehicle to move without any need for human control. These vehicles are also called driverless car or self-driving car. Advanced control systems interpret sensory information to identify appropriate navigation paths as well as obstacles and relevant sinage [12-13].

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 decides the next step in its action to avoid colliding with the obstacle.

A description of a virtual window is given along with its implementation within the context of this work. Then, the technique that is applied using two virtual windows to calculate the path of the uniformly

moving obstacle is presented. A simulation applying this technique is performed to corroborate the effectiveness of the proposed system.

2. METHODOLOGY

2.1. 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 [14]. 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 Figure 1. Assuming the speed of light in air to be 2 × 108ms-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.

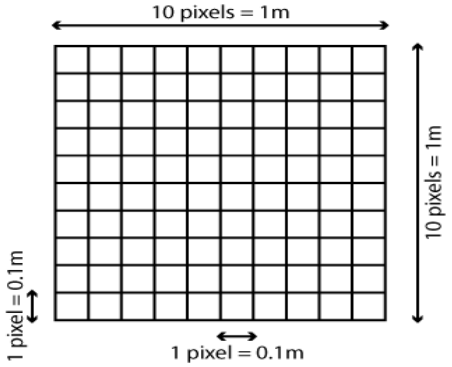

Figure 1. A visual image representation of the virtual window

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 [15-17].

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.

2.2. 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 Figure 2. Intersections with the two virtual windows will give sufficient information to calculate the path of the uniformly moving obstacle.

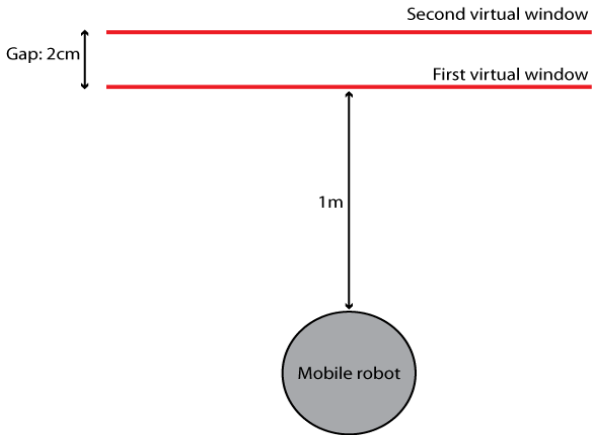

Figure 2. Top view showing the two virtual windows with respect to the mobile robot

Table 1 shows the relationship between the maximum theoretical speed of the moving obstacle that can still be detected with respect to the pixel resolution and gap between the two virtual windows. For a gap of 2cm and with a pixel resolution of 10 × 10, the maximum theoretical speed that the moving obstacle can move while still being detected by the virtual windows is about 10,000ms-1 (equivalent to 36,000kph, or 22,370mph). This does not mean that the system cannot still determine the path at a higher speed, but there will be an associated offset error with speed greater than the theoretical maximum speed. This will be shown later in a simulation. For a more typical speed of an experimental mobile robot [18], the system is very much able to detect the intersections with the two virtual windows with ample time for processing.

Similarly, the maximum theoretical speed for a gap of 2cm and pixel resolution of 100 × 100 is a more realistic 10ms-1. Maximum theoretical speeds for other values of the two parameters are shown in Table 1.

Table 1. Relationship Between Maximum Speed of Obstacle to the Pixel Resolution and Gap Between the Virtual Windows

Gap bet. virtual windows	Pixel resolution	
(cm)	10 × 10	100 × 100
2	10,000ms^{-1} (≈ 36,000kph)	10ms^{-1} (≈ 36kph)
5	25,000ms^{-1}	250ms^{-1}
10	50,000ms^{-1}	500ms^{-1}

3. SIMULATION RESULTS AND ANALYSIS

3.1. 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 Figure 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).

It is obvious from Figure 3 that the straight-line equation of the path of the obstacle is:

$$y = x + 2 \quad (1)$$

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 Figure 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 1), thus having component speeds for both x and y direction as 353.5534cms-1.

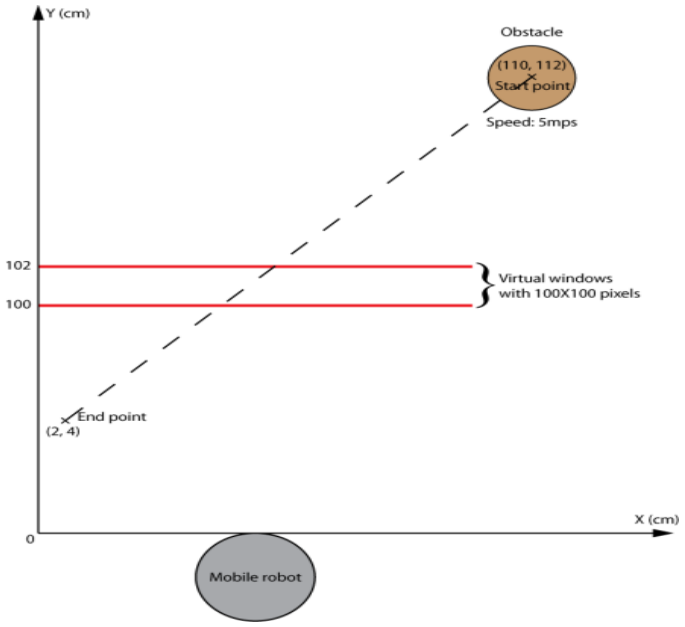

Figure 3. Top view showing relative position of the mobile robot and the obstacle, along with other relevant parameters

From Table 2, 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 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%.

Time (second)	x	y
0.0002	110	112
	353.5534	353.5534
Time (second)	x cm	y cm
0	110.0	112.0
0.000200	109.9	111.9
0.000400	109.9	111.9
0.000600	109.8	111.8
0.000800	109.7	111.7
0.001000	109.6	111.6
0.001200	109.6	111.6
0.001400	109.5	111.5
0.001600	109.4	111.4
0.001800	109.4	111.4
0.002000	109.3	111.3
0.002200	109.2	111.2
0.002400	109.2	111.2
0.002600	109.1	111.1
0.002800	109.0	111.0
0.003000	108.9	110.9
0.003200	108.9	110.9
0.003400	108.8	110.8
0.003600	108.7	110.7
0.003800	108.7	110.7
0.004000	108.6	110.6
0.004200	108.5	110.5
0.004400	108.4	110.4
0.004600	108.4	110.4

(a)

0.028200	100.03	102.03
0.028400	99.96	101.96
0.028600	99.89	101.89
0.028800	99.82	101.82
0.029000	99.75	101.75
0.029200	99.68	101.68
0.029400	99.61	101.61
0.029600	99.53	101.53
0.029800	99.46	101.46
0.030000	99.39	101.39
0.030200	99.32	101.32
0.030400	99.25	101.25
0.030600	99.18	101.18
0.030800	99.11	101.11
0.031000	99.04	101.04
0.031200	98.97	100.97
0.031400	98.90	100.90
0.031600	98.83	100.83
0.031800	98.76	100.76
0.032000	98.69	100.69
0.032200	98.62	100.62
0.032400	98.54	100.54
0.032600	98.47	100.47
0.032800	98.40	100.40
0.033000	98.33	100.33
0.033200	98.26	100.26
0.033400	98.19	100.19
0.033600	98.12	100.12
0.033800	98.05	100.05
0.034000	97.98	99.98
0.034200	97.9	99.9

(b)

Figure 4. (a) A sample of the calculations as an image captured from Microsoft Excel™ at a time step of 0.0002s, (b) A sample of the results with the two intersections highlighted and high precision simulated values

Table 2. The Position Coordinates of the Moving Obstacle in the Vicinity of First (y=100cm) and Second (y=102cm) Virtual Windows

Virtual window	Coordinates (cm)	
	x	*y*
Second	100.03	102.03
	99.96	101.96
First	98.05	100.05
	97.98	99.98

From the coordinates of intersection in Table 2, the equation of the path of the obstacle is found to be:

$$y = x + 2 \quad (2)$$

The path of the obstacle obtained through the simulation is exactly the same as the set path of the obstacle 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 error as shown above is very small, less than 0.05%.

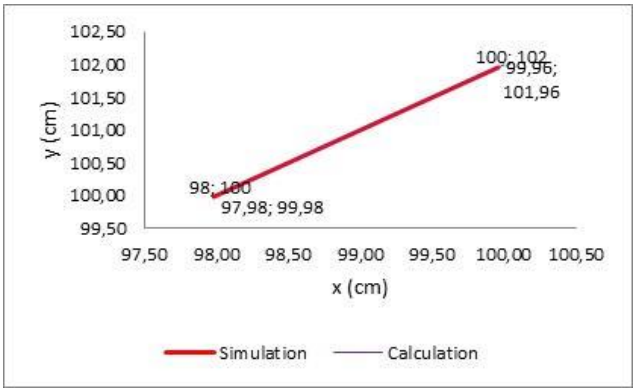

Figure 5. Comparison of the simulated path against the calculated path of the moving obstacle for a time step of 0.0002s

Alternatively, the coordinates for the two intersections with both virtual windows can be considered to be (97.98, 100) instead of (97.98, 99.98) for the first virtual window, and (99.96, 102) instead of (99.96, 101.96) for the second. These coordinates can be accepted if the system presumed that the intersections happened at y=100 for the first virtual window, and y=102 for the second virtual window even though it was detected a little bit further than the actual position of the virtual windows.

Figure 6 shows the simulated and calculated path of the moving obstacle. As can be seen, the slopes are almost similar. The slope for the simulation is found to be 1.01, thus giving a percentage error of 1%. The y-intercept, however, is found to be 1.04 and this gives a percentage error of about 48%. Even though the error of the y-intercept is large, this can be disregarded because it depends on the positioning of the axes. If the axes were placed nearer to the centre of the mobile robot, then the error is expected to be smaller, i.e. the y-intercept will be nearer to the calculated y-intercept of 2. In this study, the slope is more relevant than the y-intercept. The percentage error for the slope is just 1%.

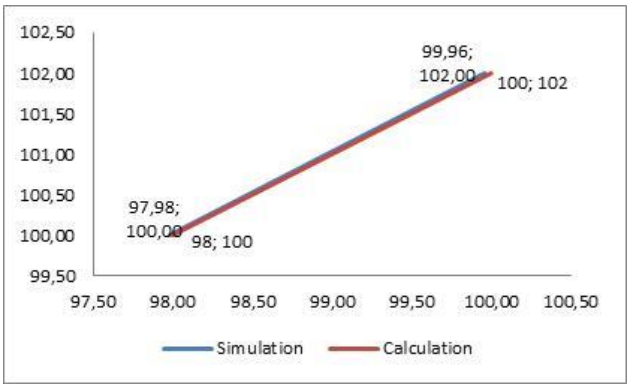

Figure 6. Comparison of the simulated path against the calculated path of the moving obstacle for a time step of 0.0002s and taking the y intersections as being 102 and 100 for the second and first virtual window respectively

Figure 7 shows another graph with the moving obstacle having a higher speed at 1000cms-1 (10ms-1). This is twice the speed of the previous exercise and at the limit of this virtual window (Table 1). The percentage errors for the slope and the y-intercept are the same as the previous exercise. This can be seen from the graph where the intersections are the same as in the previous exercise.

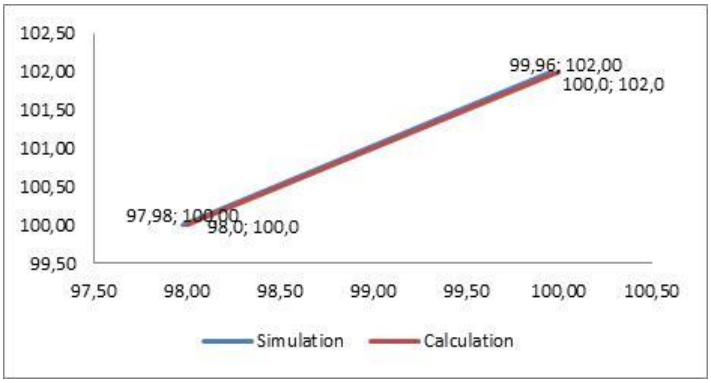

Figure 7. Comparison of the simulated path against the calculated path of the moving obstacle for a time step of 0.0002s and with obstacle speed of 10ms-1 and taking the y intersections as being 102 and 100 for the second and first virtual window respectively

Another example is performed with a higher speed of 10ms-1 (1000cms-1). Table 3 shows the point of intersections. It is the same as the obstacle with a speed of 500cms-1, and thus the percentage errors are similar too. On the other hand, if the speed is increased to 1200cms-1, the slope percentage error increased to 2%, while similarly the y-intercept percentage error is calculated to be 100%. This corroborates the limits that was calculated and shown in Table 1.

Table 3. The Position Coordinates of the Moving Obstacle in the Vicinity of First (y=100cm) and Second (y=102cm) Virtual Windows Having a Speed of 1000cms-1

Virtual window	Coordinates (cm)	
	x	*y*
Second	99.96	101.96 (102.0)
First	97.98	99.98 (100.0)

For comparison, another example is given with a time step of 0.005s. Figure 8 show a sample of the output from the simulation and Figure 9 the path plotted as a graph.

Time (second)	x	y
0.005	110	112
	353.5534	353.5534
Time (second)	x cm	y cm
0	110.0	112.0
0.005000	108.2	110.2
0.010000	106.5	108.5
0.015000	104.7	106.7
0.020000	102.9	104.9
0.025000	101.2	103.2
0.030000	99.39	101.39
0.035000	97.63	99.63
0.040000	95.9	97.9
0.045000	94.1	96.1
0.050000	92.3	94.3
0.055000	90.6	92.6
0.060000	88.8	90.8
0.065000	87.0	89.0
0.070000	85.3	87.3
0.075000	83.5	85.5
0.080000	81.7	83.7

Figure 8. A sample of the calculations as an image captured from Microsoft Excel™ at a time step of 0.005s

The slope was found to be 1.14 giving a percentage error of 14%. The y-intercept is at -11.31. This gives a percentage error of 665.4%. These values corroborates the limits that was given in Table 1.

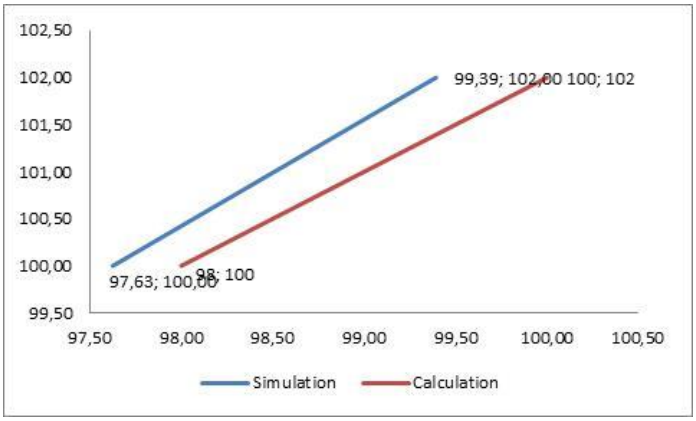

Figure 9. Comparison of the simulated path against the calculated path of the moving obstacle for a time step of 0.005s

4. 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. For the exercises did, the percentage offset error was found to be less than 0.1%.

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 Ministry of Higher Education: FRGS/1/2017/ICT04/UITM/02/5.

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*, 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*, 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] Thrun, Sebastian, "Toward Robotic Cars". *Communications of the ACM*. vol. 53 (4) pp 99–106, 2010.

[12] Lassa, Todd "The Beginning of the End of Driving". *Motor Trend*. (January 2013). Retrieved 1 September 2014.

[13] European Roadmap Smart Systems for Automated Driving, *European Technology Platform on Smart Systems Integration (EPoSS)*, 2015.

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

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

[16] *http://www.dfrobot.com/index.php?route=product/product&path=37&product_id=361*

[17] *http://www.coreconagvs.com/products/*

[18] Chung W *et al*, "High speed navigation of a mobile robot based on robot's experiences", *JSME Robotics and Mechatronics Conference*, 2006.