2013 IEEE International Conference on Control System, Computing and Engineering, 29 Nov. - 1 Dec. 2013, Penang, Malaysia

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 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.

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.

978-1-4799-1506-4 ©2013 IEEE

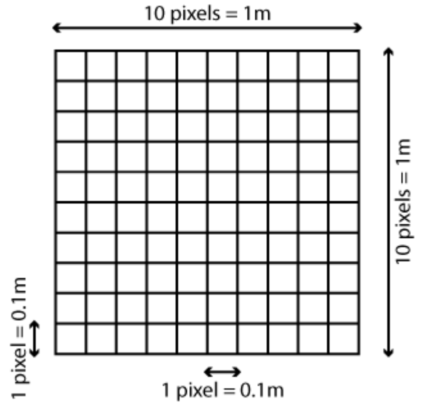


Fig. 1: A visual image representation of the virtual window.

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.

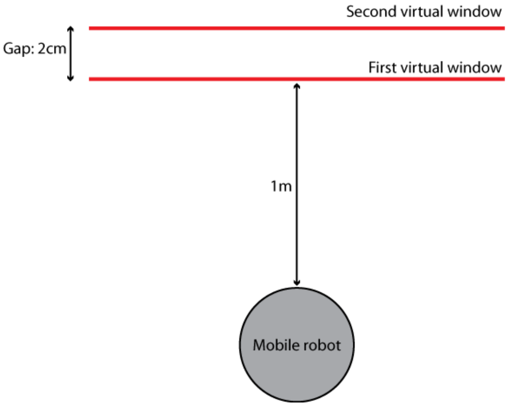


Fig. 2: Top view showing the two virtual windows with respect to the mobile robot.

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.

TABLE I. RELATIONSHIP BETWEEN MAXIMUM SPEED OF OBSTACLE TO THE PIXEL RESOLUTION AND GAP BETWEEN THE VIRTUAL WINDOWS.

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

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 $500cms^{-1}$).

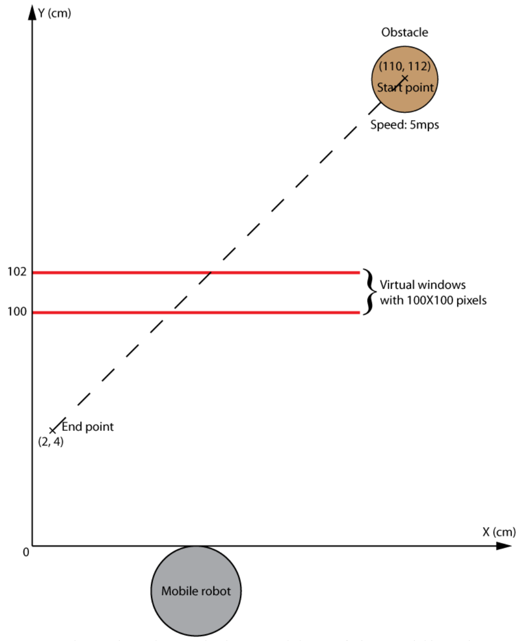


Fig. 3: Top view showing relative position of the mobile robot and the obstacle, along with other relevant parameters.

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

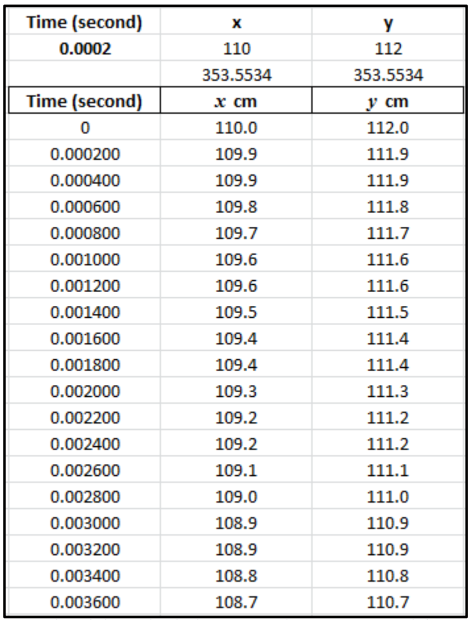
$$y = x + 2 \tag{1}$$

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}.

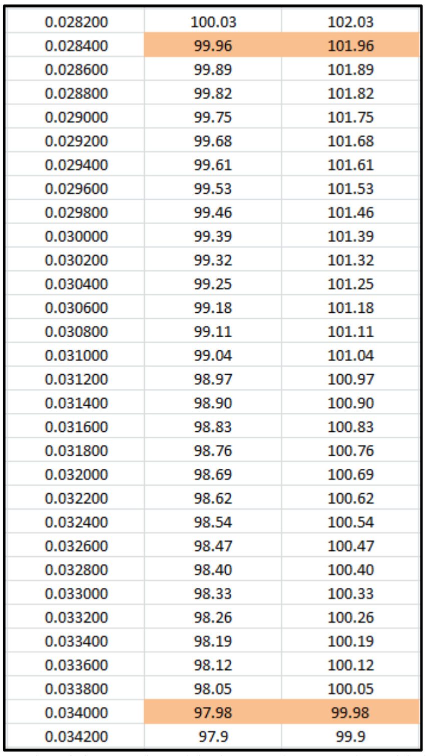
From Table II, for the second virtual window, the intersection is only detected when the location of the obstacle is at (99.4, 101.4). The intersection coordinates with the first virtual window is at (98, 100). The coordinate (100.1, 102.1) is not valid because the obstacle has not intersect the second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



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

Fig. 4a: A sample of the calculations as an image captured from Microsoft Excel™ at a time step of 0.0002s.



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

Fig. 4b: A sample of the results with the two intersections highlighted and high precision simulated values.

TABLE II. 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 II, the equation of the path of the obstacle is found to be:

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

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.

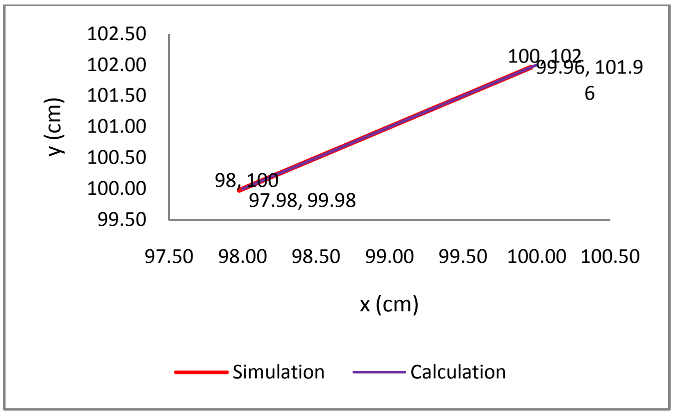
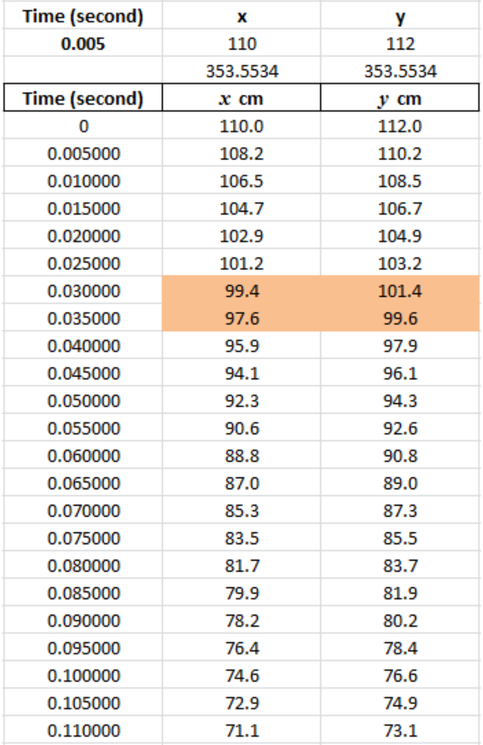


Fig. 5: Comparison of the simulated path against the calculated path of the moving obstacle for a time step of 0.0002s.

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%



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	99.4	101.4
0.035000	97.6	99.6
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
0.085000	79.9	81.9
0.090000	78.2	80.2
0.095000	76.4	78.4
0.100000	74.6	76.6
0.105000	72.9	74.9
0.110000	71.1	73.1

Fig. 6: A sample of the calculations as an image captured from Microsoft Excel™ at a time step of 0.005s.

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 happens. 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.

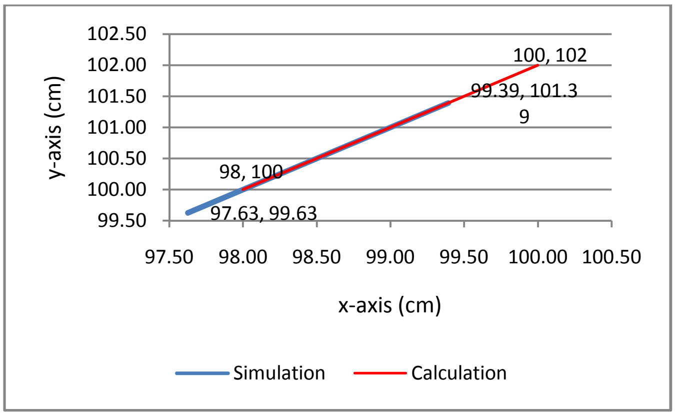


Fig. 7: Comparison of the simulated path against the calculated path of the moving obstacle for a time step of 0.005s.

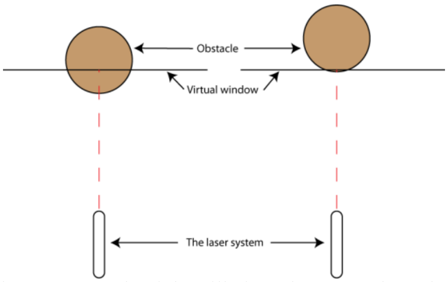
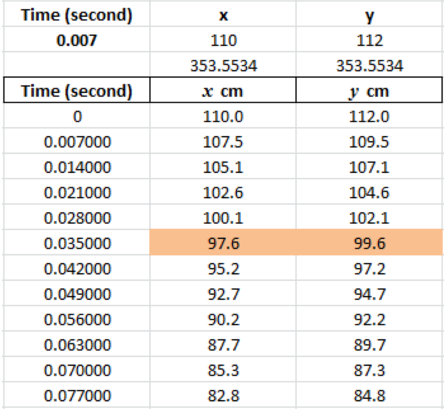


Fig. 8: The system on the right will give a better result on the point of intersection compared to the one on the left.



Time (second)	x	y
0.007	110	112
	353.5534	353.5534
Time (second)	*x* cm	*y* cm
0	110.0	112.0
0.007000	107.5	109.5
0.014000	105.1	107.1
0.021000	102.6	104.6
0.028000	100.1	102.1
0.035000	97.6	99.6
0.042000	95.2	97.2
0.049000	92.7	94.7
0.056000	90.2	92.2
0.063000	87.7	89.7
0.070000	85.3	87.3
0.077000	82.8	84.8

Fig. 9: A sample of the calculations as an image captured from Microsoft Excel™ at a time step of 0.007s.

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.

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).

TABLE III. THE POSITION COORDINATES OF THE MOVING OBSTACLE IN THE VICINITY OF FIRST (Y=100CM) AND SECOND (Y=102CM) VIRTUAL WINDOWS WITH SPEED 120MS^{-1}.

Virtual window	Coordinates (cm)	
	x	*y*
Second	101.51	103.51
	99.82	101.82
First	98.12	100.12
	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 intersection with the first virtual window. The offset percentage error with the second virtual window is 0.2%. The average offset percentage error is 0.89%.

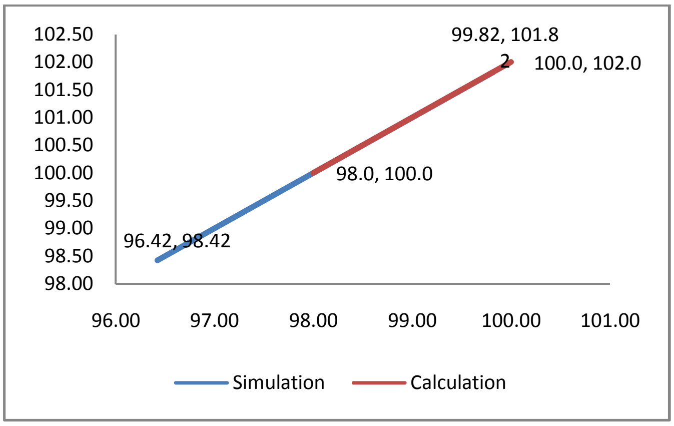


Fig. 10: Comparison of the simulated path against the calculated path of the moving obstacle with speed of 120ms^{-1}.

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&path=37&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