International Journal of Engineering & Technology, 8 (1.12) (2019) 6-10

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Path Planning Robot Using Proximity Sensor

Nurul 'Ain Amirrudin[1]*, Lim Zhen Mao[1]

[1]*Faculty of Engineering & Information Technology, MAHSA University, Bandar Saujana Putra, 42610 Jenjarom, Selangor, Malaysia*
**Corresponding author E-mail: ain@mahsa.edu.my*

Abstract

Automated robots are now commonly used in all places, from automatic production in manufacturing sites to storage arranging robots. Automated robots play an important role in our daily life. These robots may be moving in static or dynamic environment. Hence path planning has to be introduced to help robot find path while avoiding obstacles. In this research, a proximity sensor robot is modelled and simulated. The model of the robot is built and simulated in Virtual Robot Experimentation Platform (V-REP) software. A dynamic and static environment is constructed for the robot to be tested where it has to reach the end position from the start position. In static environment, all obstacles are static while dynamic environment the obstacles are movable. The performance of the proximity sensor robot is analysed in terms of the angular velocity of the robot, and the time comparison with previous research in static environment and dynamic environment. The results show that the average time taken by the proximity sensor robot is shorter than the previous research.

Keywords: *V-rep, Differential Drive Robot, Static and Dynamic environments, Path Planning, Proximity Sensor.*

1. Introduction

Path planning is robot moving from start point to end point while avoiding collisions over time [1]. Path planning is used in a major field of robotics as it gives way to autonomous vehicles. Generally, there are two types of path planning which are global path planning and local path planning. Global path planning or offline path planning refers to the path planning that can only be done where the environment is known or static. On the other hand, in local path planning or online path planning, the environment is completely unknown to the mobile robot [2-3]. This paper discusses of the reaction of path planning robot in both static and dynamic environments using V-REP path planning module.

1.1. Basic concept of wheeled mobile robots

Wheeled Mobile Robots (WMR) is a robot that consist of motorized rear wheel and at least has one orientable front steering wheel [4]. WMR is able to carry out work more efficient than legged or treaded robots on flat, smooth surface [5]. WMR consists of several mobility configurations. In this research, the WMR uses a differential drive motor where both back wheels are controlled by each motor. This enables the wheels to work independently and perform turning capability [6-7]. Besides that, there are three different types of wheels that are used for WMR; conventional wheel, omnidirectional wheel, and ball wheel. Conventional wheels are wheels that can only perform two degree of freedom (DOF) while omnidirectional wheels are wheels that has smaller wheels around its circumference which are perpendicular to the turning direction. In this research, ball wheel is used as the front wheel of the WMR because it allows the robot to navigate in three degree of freedom (DOF) while in motion.

1.2. Kinematics of the wheeled mobile robot

As mentioned, the differential drive motor robot uses two motors to control each side of the back wheel. Thus each side of the wheel has their own velocity where V_l is the left wheel's velocity of the robot and V_r is the right wheel's velocity of the robot. The differential drive motor robot can be derived with equation (1) and equation (2) [8-9]:

$$V_{rob} = (V_r + V_l)/2 \quad (1)$$

$$\omega_{rob} = (V_r - V_l)/d \quad (2)$$

where V_{rob} is the forward velocity, ω_{rob} is the rotational velocity and d is the distance between the two wheels of the robot. This enable the robot to perform trajectory tracking [10].

Copyright © 2019 Authors. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

1.3. Principle of path planning

There are four critical components that are required by mobile robot to perform navigation: sensing, environmental mapping, localisation and path planning. Path planning of mobile robot involves the finding a collision – free path from starting point to the ending point [1]. Path planning of mobile robot has been used for navigation instead of GPS due to the fact that GPS lack accuracy when the signal is weak [11]. There are two types of path planning which are global path planning and local path planning. Global path planning refers to the path planning that can only be done in an already known or static environment while local path planning is path planning in an unknown or dynamic environment [1].

2. Design of differential robot

The differential robot is designed by using the CAD data provided by V-REP. A single simple shape was imported and is located in the middle of the scene. The shape also appears in the scene hierarchy. Figure 1 shows the imported shape.

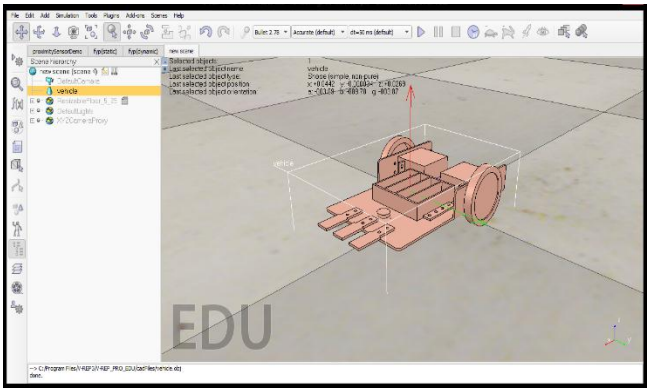


Fig. 1: The imported shape from V-REP

Steps for assembling the robot in V-Rep are summarized as follows:

- Import .obj files into V-REP.
- Divide imported object into sub parts.
- Extract the shape of the three-cuboids from main robot body.
- Extract inner wheel shape from whole wheel.
- Recolour whole robot with different colour.
- Create pure shape for battery casing of the robot.
- Repeat same steps for all other part of the robot.
- Merge all non-pure parts of the robot.
- Align all merged grouped parts with world frame.
- Attach grouped parts to their pure counterparts.
- Add revolute joints to center of both wheels.
- Add force sensor to front wheel of the robot.
- Attach threaded child script to robot.

To enable the robot is able to dynamically simulate, pure shapes has to be added to ensure the calculation time and results are always constant. The battery casing in the middle of the vehicle is selected. Also, the pure cuboid shape is added to the scene which surround the selected battery casing. The created cuboid is respondable and dynamic. This means the robot will collapse and react to collision during simulation. Once all components are selected, these components are merged together in order to form a robot. The final product of the differential motor robot is shown in Figure 2.

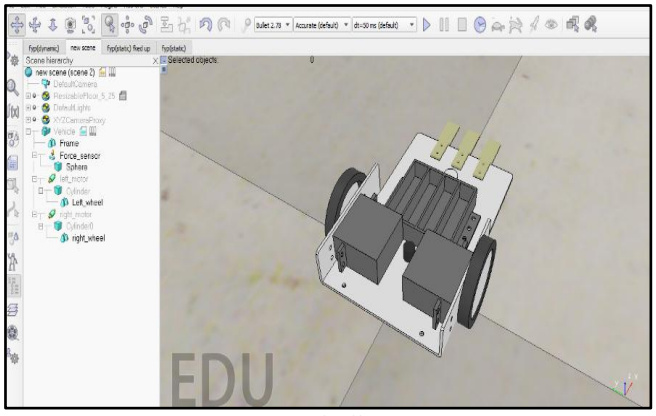


Fig. 2: Final product of differential motor robot

2.1. Proximity Sensor Robot

After the assembly of the robot parts are arranged, proximity sensor is added to detect distance between obstacle and robot. Proximity sensors are basically 'eyes' for the robot to sense its surrounding. The sensor detects the distance of the robot with the obstacle by sending out ultrasonic rays. When the ray hits an object, it is reflected back to the receiver and the distance between both robot and obstacle is obtained. In this research, proximity sensor is selected based on the radius of detection, speed of detection, precision of detection and type of detection volumes [12]. **Error! Reference source not found.** shows the properties of six considered sensors, which are ray-type sensor, randomized ray-type sensor, pyramid-type sensor, cylinder-type sensor, disk-type sensor and cone-type sensor.
Ray-type sensor has a ray-shaped volume which is a long straight line sensor ray. Due to its low radius of detection, it is unable to detect all direction of obstacle making it ineffective in path planning purpose. Randomized ray-type sensor and Cone-type sensor have the same cone-shaped volume but randomized ray-type sensor is unable to change its radius. And since its radius is already wide, it is more likely for it to pick up obstacles from the side cause more delays before reaching the destination. Besides that, Randomized ray-type sensor is unable to adjust face counts unlike cone-type sensor. Face counts are the number of line detector on the sensor, meaning the more face counts a sensor has the more precise it can be. Therefore, Cone-type sensor is preferred for this research. Pyramid-typed sensor face the same issue with randomized ray-typed sensor where face counts are un-adjustable. This renders the sensor to be not suitable for the research.

3. Simulation scenario

In this research, the performance of proximity sensor robot is studied in both static and dynamic environments. The scene of a static environment and dynamic environment are shown in Figure 3 and Figure 4, respectively. In static environment, it is assuming that all surroundings are known by robot and obstacles in the scene is stationary. On the other hand, the surroundings in dynamic environment is also assumed to be known by the robot but there are some obstacles which will rotate 90 degrees and block the route. The target destination in the scene can be moved around.

Table 1: Six proximity sensors and their properties [12]

Parameters	Ray-type sensor	Randomized ray-type sensor	Pyramid-type sensor	Cylinder-type sensor	Disk-type sensor	Cone-type sensor
Radius of detection	Very low	Very High	High	High	High	High
Speed of detection	Very fast	Fast	Fast	Fast	Moderate	Moderate
Precision of detection	Low	Moderate	Moderate	Moderate	High	Very High
Type of detection volumes	Ray-shaped volumes	Cone-shaped volumes	Rectangular-shaped volumes	Revolute-shaped volumes	Disc-shaped volume	Cone-shaped volumes

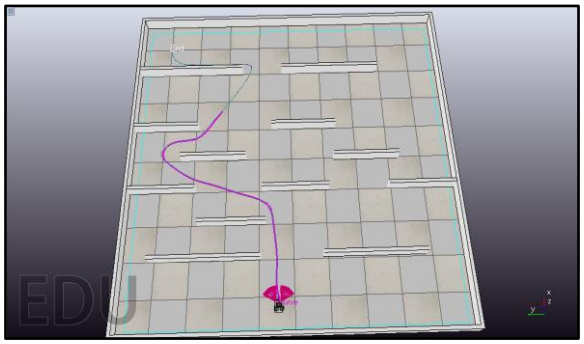


Fig. 3: The static-environment of V-REP

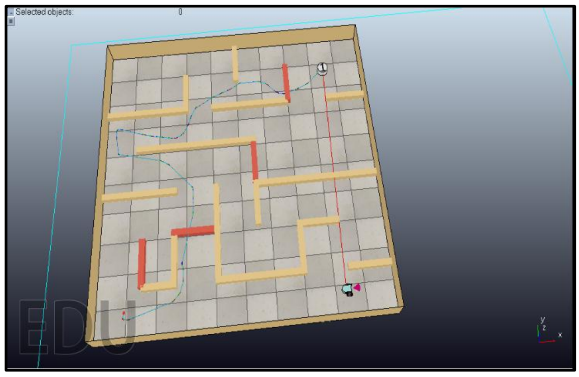


Fig. 4: The dynamic environment in V-REP

In static environment, the obstacles are walls that are added to the scene from [Model browser → Infrastructure → walls → 20cm high walls]. The position and the orientation of all obstacles are set. Then, the item under 'object special properties' must be checked to ensure that the RRT algorithm is able to take account the obstacles while creating a path.

The next step is to add a start dummy by selecting [Menu bar → Add → Dummy], rename it to 'Start' and translate it to the robot position and orientation. The start dummy would be the path point on the path for the robot to calculate its rotational velocity, ω_{rob}. Then, copy the Start dummy and translate it to a desired location in the scene. The copied dummy is renamed as End. This will be the destination of the robot. Next, a path between Start and End is set. This path will be used for the path planning module. Under path planning dialog in the module calculation properties, the Start dummy, End dummy, and path are defined. A blue square is visible around the robot. This indicates the search parameters to find path between Start dummy and End dummy. After the path planning module is finished, set the robot as the child of the Start dummy and click on compute path. A path will be produced that would avoid all obstacles. The path may be different for each time the simulation made.
In dynamic environment, a moving obstacle is added. This obstacle is always moving during the simulation and can block the route. Whenever the obstacle blocks the route, the previous path will be changed to a new path immediately. Thus, the robot will not be blocked by the obstacle and has a smooth path until it reaches the destination. The position of the obstacle is random. Thus, the path of the robot will be varied for each time the simulation made.

4. Result and discussion

The performance of the proximity sensor robot is analysed in terms of the angular velocity of the robot, and the time comparison with previous research in static environment and dynamic environment.

4.1. Angular velocity of the robot

Figure 5 shows the angular velocity of the path planning robot. The graph started off with 0 degree per second as the robot is still stationary. Angular velocity or rotational velocity is defined as per equation (3) [13]:

$$V_{wheel} = \omega_{wheel} * R_{wheel} \tag{3}$$

where ω_{wheel} is the rotation rate of the wheel, R_{wheel} is the radius of the wheel, and V_{wheel} is the velocity of the wheels. Each time the robot is making a turn at a curve, the angular velocity increases dramatically. This is because when the robot is turning in a curve, the rotation rate of the wheel increases causing the velocity of the wheels to increase as well. When robot returns to straight path, its rotational rate decreases leading to the decrease of the velocity of the wheels. The highest reading is 57.41 degree per second at 55.55s. After the robot reaches to its destination, the angular velocity drops to zero.

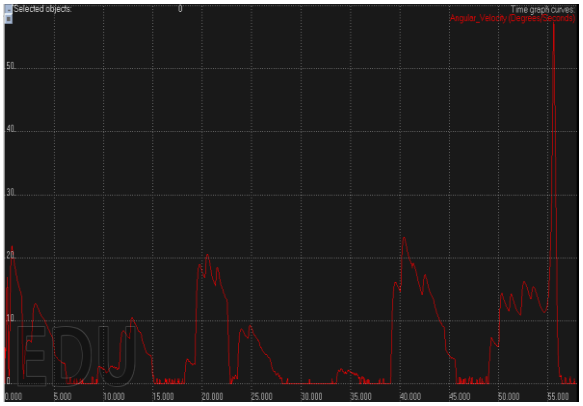


Fig. 5: Angular Velocity of the Robot

4.2. Time comparison in a static environment

Time comparison between proximity sensor robot and force sensor robot [14] in a static environment has been conducted in this research. Five samples have been collected and recorded. Figure 6 shows the Path Taken by Robot vs Time Taken to Reach Destination Chart.

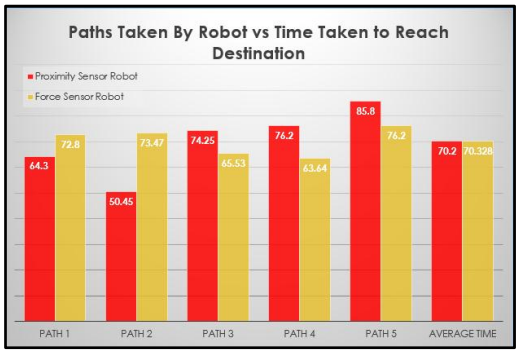


Fig. 6: Path Taken by Robot vs Time Taken to Reach Destination Chart in Static

The chart shows that there is not much difference in average time for the proximity sensor robot and force sensor robot. The longest time taken by the proximity sensor robot was 85.8s while force sensor robot took 76.2s. Since path planning module is meant to avoid obstacles assuming the environment is already known. Hence, no collision occurred in static environment since all obstacle has been calculated and

avoided. Each time the simulation runs, the path produced by RRT is randomized. Therefore, the time taken for each path by both robots have significant difference. In this research, an improvement has been done by using the proximity sensor to detect the distance between the robot and the obstacle.

4.3. Time comparison in a dynamic environment

Time comparison between proximity sensor robot and force sensor robot [14] in a dynamic environment has been conducted in this research. Five samples have been collected and recorded. Figure 7 shows the Path Taken by Robot vs Time Taken to Reach Destination Chart.

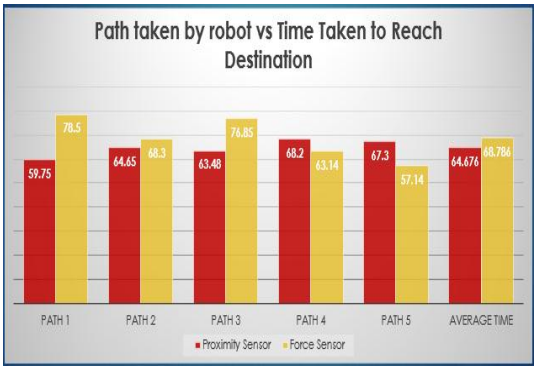


Fig. 7: Path Taken by Robot vs Time Taken to Reach Destination Chart in Dynamic

The chart shows the result that is compared between proximity sensor robot and force sensor robot. Based on the average time, the proximity sensor robot takes a shorter time to reach to its destination which is 64.676 s compared to the force sensor which is 68.786 s. This is because the force sensor robot detects obstacles by colliding into them. When collision occurred, the robot moves backwards and recalculates a new path. Thus, it will increase the time taken. However, the proximity sensor robot is able to detect the obstacle from a distance before the robot collides into it. This small changes have save up a lot of time for the robot to reach its destination especially when there is much moving obstacles along the path.

5. Conclusion

Proximity sensor robot is designed by using V-REP software. Two environments are considered which are static environment and dynamic environment. Path panning module in V-REP software has been used to calculate the paths in both environments.
The performance of the proximity sensor robot is analysed and compared with force sensor robot [1]. The results show that there is not much difference in average time for both robots in static environment. However, there is some improvement in dynamic environment where the proximity sensor robot takes a shorter time to reach to its destination compared to the force sensor. This is because the proximity sensor robot is able to detect the obstacle from a distance before the robot collides into it.

Acknowledgement

The authors would like to convey gratitude to the lecturers in Faculty of Engineering and IT, MAHSA University on the knowledge given and support to accomplish this project, and provide a convenient lab to do the simulation on V-REP.

References

[1] T.Lv, C.Zhao, and J.Bao, "A Global Path Planning Algorithm Based on Bidirectional SVGA", *Journal of Robotics*, Vol. 2017, (2016).
[2] Leena, N., and K. K. Saju. "A Survey on Path Planning Techniques for Autonomous Mobile Robots", *IOSR Journal of Mechanical and Civil Engineering (IOSR-JMCE)*, Vol. 2014, (2014), pp. 76–79.
[3] M. Samadi and M. F. Othman, "Global Path Planning for Autonomous Mobile Robot Using Genetic Algorithm", *Signal-Image Technology & Internet-Based Systems (SITIS), 2013 International Conference on*. IEEE, 2013, pp. 726–730.
[4] Morin, Pascal, and Claude Samson, "Motion Control of Wheeled Mobile Robots", *Springer Handbook of Robotics*, Springer (2008), pp:799-826.
[5] De Luca, Alessandro, Giuseppe Oriolo, and Marilena Vendittelli, "Control of Wheeled Mobile Robots: An Experimental Overview", *Ramsete*, Springer (2001), pp:181–226, 2001.
[6] Ortigoza, Ramon Silva, et al, "Wheeled Mobile Robots: A Review", *IEEE Latin America Transactions*, Vol. 10, No. 6, (2012), pp: 2209–2217.
[7] Delgado-Mata, Carlos, Ramiro Velázquez, and Carlos A. Gutiérrez, "A Differential-Drive Mobile Robot Driven by an Ethology Inspired Behaviour Architecture", *Procedia Technology*, Vol. 3, (2012), pp:157–166.
[8] Malu, Sandeep Kumar, and Jharna Majumdar, "Kinematics, Localization and Control of Differential Drive Mobile Robot", *Global Journal of Research in Engineering*, Vol. 14, No. 1, (2014), pp:1–8.
[9] Myint, Cherry, and Nu Nu Win, "Position and Velocity Control for Two-Wheel Differential Drive Mobile Robot", *International Journal of Science, Engineering and Technology Research (IJSETR)*, Vol. 5, No. 9, (2016), pp:2849–2855.
[10] Robins Mathew and S. S. Hiremath, "Trajectory Tracking and Control of Differential Drive Robot for Predefined Regular Geometrical Path," *Procedia Technology*, Vol. 25, (2016), pp:1273–1280.
[11] Mohammed Z. Al-Faiz and Ghufran E. Mahameda, "GPS-based Navigated Autonomous Robot", *International Journal of Emerging Trends in Engineering Research*, Vol. 3, No. 4 (2015), pp:8–13.
[12] Coppelia Robotics, "Proximity Sensor Types and Mode of Operation", *Virtual Robot Experimentation Platform User Manual*, (2017), available online: http://www.coppeliarobotics.com/helpFiles/en/proximitySensorDescription.htm.
[13] Leena N. and K. K. Saju, "Modelling and Trajectory Tracking of Wheeled Mobile Robots", *Procedia Technology*, Vol. 24, (2016), pp:538–545.
[14] Samuel Abhishek and V.Kavati, "Trajectory Planning of a Mobile Robot", *National Conference on Technological Advancements in Mechanical Engineering*, (2016).