Proceedings of the 3rd International Conference on Signal Processing and Machine Learning
DOI: 10.54254/2755-2721/5/20230639

Global and local path planning for mass point robots in two-dimensional environments

Yao Yang[1]

[1]Yancheng Experiment High School, Yancheng, 224000, China

2019010070@stu.cdut.edu.cn

Abstract. Robot path planning is a popular research topic due to its wide application. There are various approaches to apply to path planning, such as artificial potentials-based approach, probability-based approach, retraction-based approach, and heuristic approach. In this paper, the heuristic approach is focused on due to its convenience. In heuristic approach, considering the application, the author chooses fuzzy logic and neural network to apply to path planning. By conducting experiments both on PowerBot and the simulation environment, the methods proposed in the article are feasible. The environment data is collected by the sensors. Next, the fuzzy controller or neural network will process the data, and generate an appropriate path to get to the target without hitting any obstacles or walls.

Keywords: Robot path planning, Heuristic approach, Fuzzy logic, Neural network.

1. Introduction

Path planning is an algorithm to find a path from the initial point to the target. It has a wide application range, for example, sweeping robots, hospital transferring robots and mobile picking robots in warehouses. In this paper, the author will discuss the application of heuristic approaches in robot path planning.

Robots must employ global or local route planning in various conditions. A global route planner creates a low-resolution, high-level path on a map. Local route planning requires no environmental background. It delivers a high-resolution low-level route based on onboard sensor data. [1]. There are also many approaches related to path planning, such as the artificial potentials-based approach, probability-based approach, retraction-based approach, rolling window approach, Dijkstra approach and heuristic approach. The artificial potentials-based method approach consists mostly of applying potential fields around configuration-space obstacles [2]. In a probability-based approach, a random node is generated in the configuration space [3]. Retraction-based approach retracts a sample or a configuration to a more desirable region of the free space [4]. In the rolling window technique, it converts uncertain local environment data to a recognised range [5]. In the Dijkstra approach, the adjacency matrix is employed as the approach's naive storage structure [6]. In the heuristic approach, roughly estimated data on how far the robot is from the target. By analysing the benefits and setbacks of the approaches above, the heuristic approach is the most suitable one. It only needs to sense the surroundings to get the estimated data and needs low computation without doing a lot of trials and inputting the environment map in advance. As a result, heuristic approaches are selected for the subject of this article.

DOI: 10.54254/2755-2721/5/20230639

This article describes the use of two kinds of heuristic approaches. They are fuzzy logic techniques and neural networks. The fuzzy logic technique is a path planning algorithm stimulated by subjective uncertainty in the human mind. As humans can navigate themselves with the aid of rough data, researchers are inspired by this to create the fuzzy logic technique. This algorithm applies a set of IF-THEN rules to help the robot make decisions. The use of neural networks for robot route planning has allowed for the stimulation of the connection that exists between the inputs and the outputs. According to [1], the use of neural networks in robot navigation may be broken down into three distinct categories: the first of these is interpreting the sensory input, the second is avoiding obstacles, and the third is planning routes.

In this paper, the fuzzy logic technique is discussed in 2.1. It mainly includes decomposing the path planning into different behaviours by using linguistic rules [7], and the Degree of applicability (DOA) [8]. Neural network is discussed in 2.2. Applications of neural networks in robot path planning will be explained. Authors in [9] proposed a four-layer neural network in global path planning, and [10] devised a dynamic neural network, and explain the details of training the neural network. In Section 3, the result is presented with the aid of some experiments. In Section 4, the reference is listed.

2. Methods

In this section, the research consists of fuzzy logic techniques and neural networks. They are two of the widespread heuristic approaches.

2.1. Fuzzy logic technique

The approach of fuzzy logic is especially well suited for the implementation of tasks that do not need accurate measurements or calculations because of its skills of inference and approximate reasoning under uncertainty. This is because fuzzy logic can reason approximately. The approach of fuzzy logic has been presented for use in a broad variety of contexts by a large number of scholars and scientists.

It has been recommended that a systematic methodology be devised for dividing the process of route planning into different behaviours and carrying each one out either alone or in conjunction with the others. Each behaviour has its own unique collection of fuzzy rule statements that make up the behaviour.

The whole of the difficult behaviour will be broken down into a number of different behaviour modules that are more basic in an architecture that is known as subsumption. The behaviours of obstacle avoidance, wall following, and emergency are shown in ‘figure 1’, and mobile robot tasks often call for all three of these behaviours to be present. Other actions, such as adhering to the outlined plan of conduct and accomplishing one's objectives, could also be required. A fuzzy system that provides linguistic rules for each behaviour might be useful in finding a solution to the issue of behaviour arbitration. Each behaviour is represented by a fuzzy logic rule foundation. This strategy is not only beneficial in establishing the rules for each behaviour, but it is also helpful in doing so. It's possible that the task supervisor will utilise sensor inputs to figure out how the robot will behave. This gives the robot a greater degree of independence. Choices made by humans are possible when using programmable manual control. The actions of the robot are determined by its tasks. In this research, all behaviours will be carried out with the exception of wall-following, which will be carried out using a stereovision camera. The modular structure of the fuzzy rule foundation for route following, objective completion, and obstacle avoidance will be explained in the following paragraphs.

Wall-following behaviour minimises orientation error alpha. Difference between the intended and actual direction. Similarly, one of our aims is to lower the robot's distance error d

$$d = \sqrt{(x_t - x_r)^2 + (y_t - y_r)^2} \tag{1}$$

$$\alpha = \varphi - \operatorname{atan}2\big((y_t - y_r), (x_t - x_r)\big) \tag{2}$$

The distance d can be defined by linguistic variables as Z-Zero, NZ-Near Zero, N- Near, M- Medium, NF- Near Far and F- Far. In Obstacle avoidance behavior, the distance linguistic value of d_F is defined as $max(\mathrm{dF}_1, \mathrm{dF}_2, \mathrm{dF}_3)$, the distance linguistic value of d_L is defined as $max(\mathrm{dL}_1, \mathrm{dL}_2, \mathrm{dL}_3, \mathrm{dL}_4)$, and the distance linguistic value of d_R is defined as $max(\mathrm{dR}_1, \mathrm{dR}_2, \mathrm{dR}_3, \mathrm{dR}_4)$.

DOI: 10.54254/2755-2721/5/20230639

The behaviour in an emergency situation is given the utmost attention. When the robot approaches a potentially dangerous area where it is more likely to sustain mechanical harm, it will activate this feature. The user is responsible for determining the emergency distance that must be maintained by the robot in order to remain inside the safe navigation zone. The behaviour is as follows

Algorithm 1

Specify d_e
Get sensed distances: d_L, d_F, *and* d_R
  If ($d_L < d_e$ *OR* $d_F < d_e$, *OR* $d_R < d_e$)
    Then STOP
  Else CONTINUE
  EndIf

Primitive behaviours are straightforward, low-level behaviours that often accept input from the sensors of the robot and then translate that information to the actuators of the robot in a nonlinear fashion. Some behaviours are composites. DOA is a measurement of how strongly active it suddenly is. It is necessary to mix the DOA-weighted fundamental behaviours in order to produce composite behaviours. The fusion of behaviours degenerates into a switching of behaviours when the degree of overlap between those behaviours is either zero or one.

It is possible to arrive at inferences about meanings thanks to fuzzy algorithms, which use weights as an integral element of their operation. Fuzzy algorithms have made it easier for humans to control and operate robots in recent years. Before a fuzzy inference system can be developed, there are a number of phases that need to be completed, two of which are the construction of fuzzy logic and the assignment of command weights. The process of putting together a fuzzy inference system requires many steps. Figure 2, which may be viewed over here, provides an illustration of the framework of a fuzzy inference system. It is outfitted with a fuzzy inference system that stores data on the present condition and standing of a mobile robot. As a direct result of this, the system is now in a position to provide an image of the situation that is more accurate. One is possible to form an informed prediction about the control and rotational velocity cost function by reviewing the weights that were generated based on the obtained data.

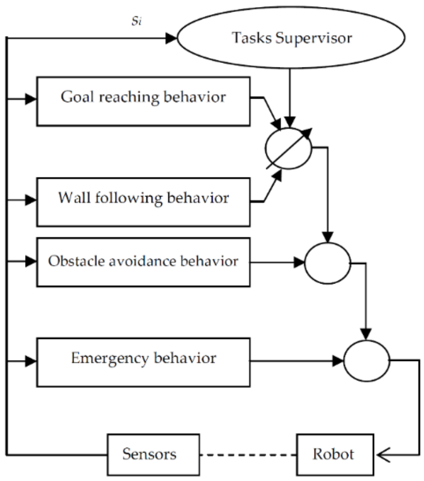

Figure 1. Behavioural-based fuzzy control architecture[7].

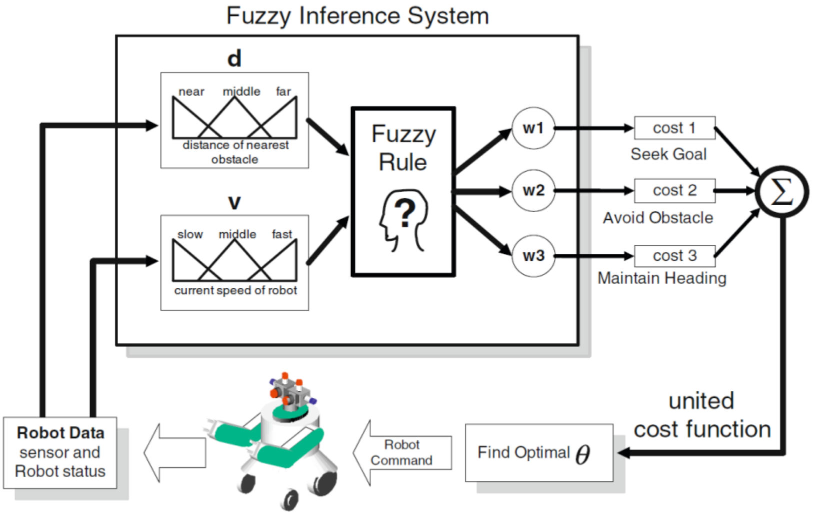

Figure 2. Behavioural-based fuzzy control architecture [8].

2.2. Neural network

The following strategies have been offered as potential ways to incorporate neural networks into the process of designing robot routes. It has been suggested that a neural network with four layers be used to calculate the angle at which the robot is pointing as it moves in the direction of a goal that has been predetermined. This has been advocated. In a neural network with four layers, there are three input neurons for each of the three possible directions in which a robot may travel in relation to obstacles. Target receives one neuron. The steering angle of the movement is generated by a single neuron in the output layer. Because it's so difficult to train neural networks, they include two layers of hidden data.

It's possible that using this neural network to design routes all around the globe would be helpful. The purpose of this neural network is to offer mobile robots collision-free motion planning so that the robots may carry out the tasks that have been assigned to them. During the process of navigation, the robot has to keep a safe and suitable distance to obstacles, improve its mobility, decrease the amount of time it takes for it to travel, and avoid both static and dynamic barriers.

The input values include the left obstacle distance from the robot $y_1^{\{1\}}$, the front obstacle distance from the robot $y_2^{\{1\}}$, the right obstacle distance from the robot $y_3^{\{1\}}$, and the target bearing of the robot $y_4^{\{1\}}$. Then, the input values generate outputs as following

$$y_2^{\{\text{lay}\}} = f\left(v_j^{\{\text{lay}\}}\right) \tag{3}$$

where

$$V_j^{\{\text{lay}\}} = \sum_i W_{ji}^{\{\text{lay}\}} \cdot y_i^{\{\text{lay}-1\}} \tag{4}$$

The gradient for neurons in $\{lay\}$ shows as following

$$\delta_j^{\{\text{lay}\}} = f'\left(V_j^{\{\text{lay}\}}\right)\left(\sum_k \delta_k^{\{\text{lay}+1\}} W_{kj}^{\{\text{lay}+1\}}\right). \tag{5}$$

The final output is:

$$\theta_{actual} = f(\mathcal{V}_1^4) \tag{6}$$

where

$$V_1^4 = \sum_i W_{1i}^{\{4\}} y_i^{\{3\}}. \tag{7}$$

DOI: 10.54254/2755-2721/5/20230639

In certain instances, the learning process could even take place during the behaviour of "target-seeking," depending on the circumstances. It gives the impression that the design of routes may have to adjust to account for changes in the environment as they occur.

The result is validated via the use of simulations and tests carried out with the assistance of the mobile robot KHEPERA-III. During each of the audits, the robots can go to locations that were allotted to them in a timely manner. This was confirmed by the findings of the inspections. In terms of precise positioning and avoiding collisions, the neural controller robot performed far better than its fuzzy controller counterpart.

The foundation is laid for the development of a dynamic neural network. The quantity of accessible space acts as the system's input, while the steering angle and velocities emerge from the system as the system's output. This particular neural network can operate correctly without any form of environment map being present. During the course of the programme, students will engage in both conventional in-person classroom teaching as well as online study. A method called particle swarm optimization is utilised in order to build a path that is as smooth as possible. A dataset that is not optimal is subjected to processing that is conducted automatically.

The training has two stages. The first training stage is offline training. d_j is the distance between the robot and the obstacle at sector j. β_j is the orientation of the j^{th} sector, S_j, with respect to the local x-axis.

The second component of the course is instruction that is provided via the use of the internet. The turning radius that yields the greatest outcomes is a mystery at the beginning of the process. In spite of this, it is feasible to estimate how far the robot's radius extends into the surrounding area. At this stage in the training process, the network will get feedback on its performance based on an evaluation function. This input will be provided in order to help the network improve. What you will see when you look at the evaluation function is as following

$$F(k) = \frac{1}{2}e^2(k) = \frac{1}{2}\left(e_x^2(k) + e_y^2(k)\right) \tag{8}$$

with

$$e_x(k) = x_t(k) - x(k) \tag{9}$$

$$e_y(k) = y_t(k) - y(k) \tag{10}$$

where x_t, y_t are the coordinates of target Cartesian and x, y are the coordinates of current robot Cartesian.

The weight is modified according to

$$\Delta W = -\eta\frac{\partial F(k)}{\partial W} \tag{11}$$

$$W(k+1) = W(k) - \eta\frac{\partial F(k)}{\partial W} \tag{12}$$

Following the training above, the neural network functions well in robot path planning. In [11] and [12], it is shown that the planning algorithms can be applied to multi-target tracking applications for teams of robots.

3. Results

To test the feasibility of the method, the author uses two experiments. PowerBot is employed in the first experiment [3]. Random-Obstacles Scenario scatters obstacles throughout the testbed. The Fuzzy-Logic motion controller (FLMC) should make the path to its target without colliding with any impediments. Figure 3 depicts the mobile robot's development amid static barriers, while figure 4 illustrates its velocity profile.

DOI: 10.54254/2755-2721/5/20230639

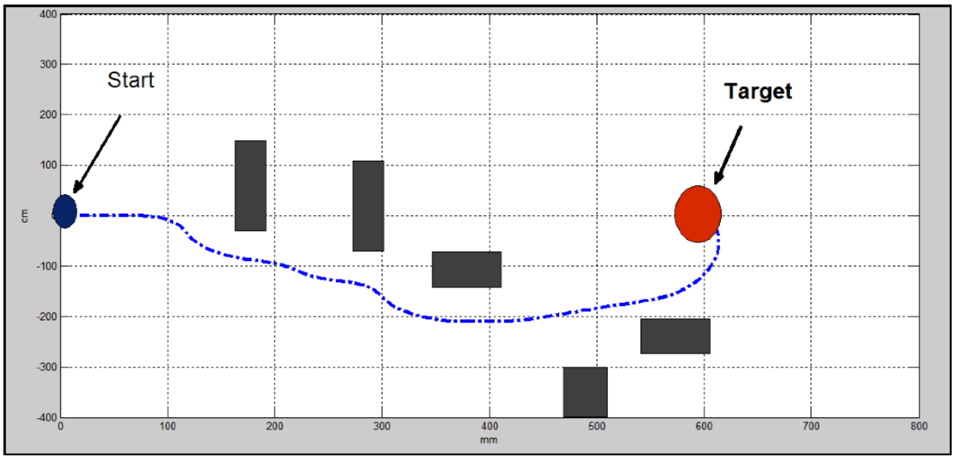

Figure 3. Random-Obstacles Scenario[7].

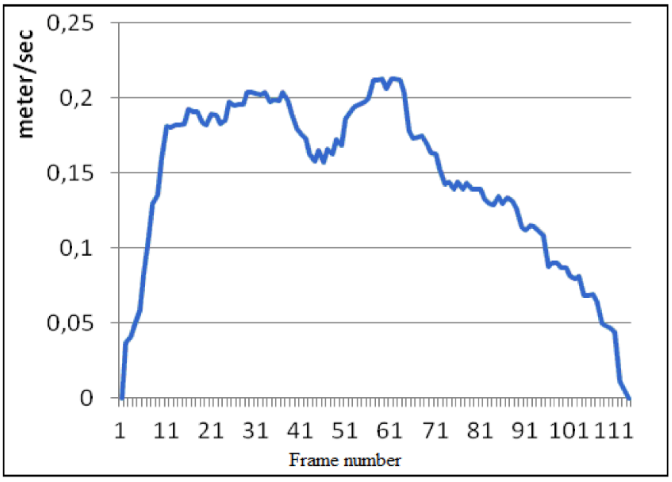

Figure 4. Profile of the velocity for the random obstacles scenario [7].

Because the obstacles have been strategically positioned to prevent the robot from entering a dead-end situation, the robot travels smoothly, avoids obstacles, and achieves the target through an optimum or sub-optimal route without aid from the stereovision path-planning module.

In the Random Obstacles Scenario, roadblocks are randomly placed throughout the test environment. Now that the robot is not confined, it will enter the cul-de-sac even. This helps the robot swiftly navigate away from busy locations. It will take more time and be less efficient to escape if you rely only on the fuzzy-logic motion controller (FLMC).

Figure 5 depicts the course travelled by the mobile robot as a consequence of the FLMC and SVPPM. The robot enters the Cul-de-Sac situation at any time during execution through an optimum path, accomplishes the destination using an optimal path created by the SVPPM, and avoids obstacles. An illustration of the matching velocity profile is given in figure 6. On a grid map that was constructed via stereovision-based SLAM, figure 7 displays the route that was generated by the SVPPMM for the situation that was described earlier. This path was developed for the purpose of the scenario that was mentioned previously.

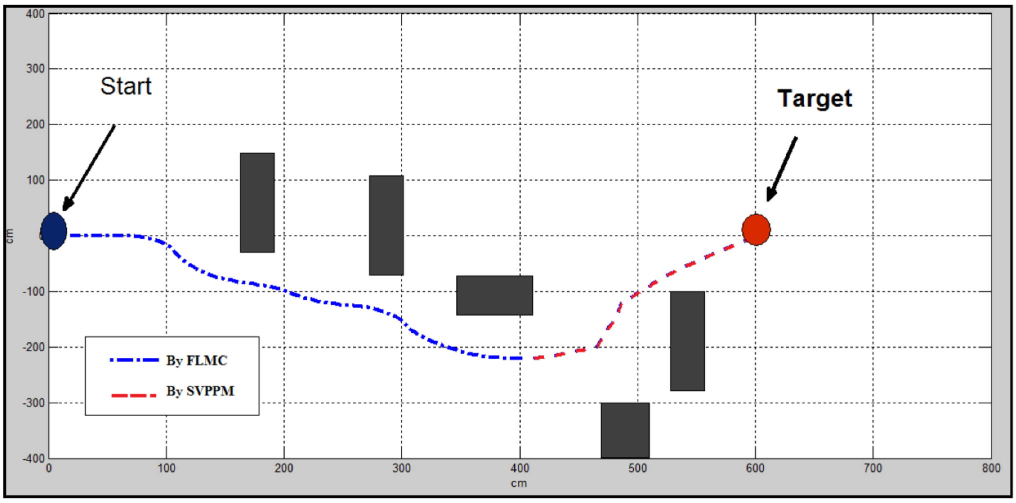

Figure 5. Cul-de-Sac with in Random Obstacles Scenario [7].

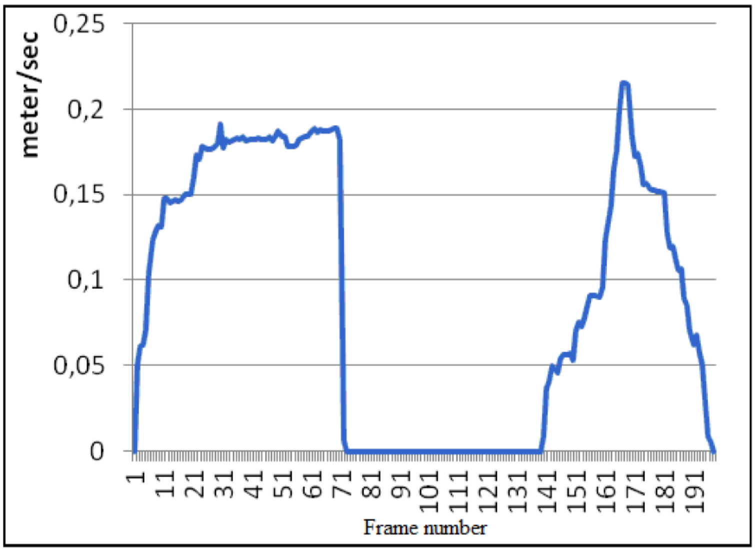

Figure 6. Profile of the velocity in a Cul-de-Sac with Random Obstacles Scenario [7].

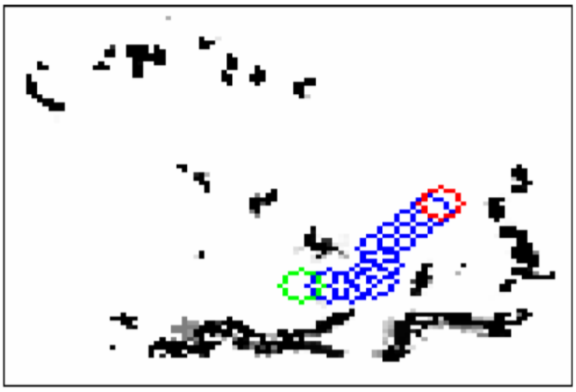

Figure 7. SVPPM generated path on a grid map [7].

DOI: 10.54254/2755-2721/5/20230639

The second experiment [5] uses ROBNAV software based on C++. Visual behaviour in navigation is guided by a person's perception of motion. The presence of visual motion provides additional context to the surroundings. Robots need good obstacle avoidance capabilities in order to function properly. Static barriers may take the form of a variety of different things, including but not limited to walls, poles, fences, trees, and other items. Moving obstructions include vehicles, people, animals, etc. People's reactions to these types of things could trigger avoidance behaviours in them, such as slowing down, making a turn, or coming to a full stop. As a result, the robot is able to ascertain whether or not an item is near enough to it to cause a collision. The robot will move in the opposite direction in order to avoid colliding with anything that is too near to it. Figure 8 shows that the primary reactive behaviour is decelerating in order to escape both static and moving obstacles. Follow walls between rooms. The mobile robot enters wall-following mode when it finds a barrier in its path. The mobile robot recognises a front obstruction in target tracking mode. When it reaches U-shaped or dead-end obstructions, the robot can't achieve its target without wall following. In this situation, the robot should keep going. The robot is getting closer to obstructions as it moves forward. When attempting to avoid obstacles, any behaviour other than "wall following" would cause a split. For the robot to see, locate, and approach the goal shown in figure 9(a) and avoid becoming stuck in the trap depicted in figure 9(b), the edge behaviour must be as follows.

Target search assumes a mobile robot's goal is fixed. In the case of getting from here to there through a network of paths, the objective location is fixed. The robot target-steers when sensors detect no obstacles. Figure 8 show how the controller alters directions. Neural networks teach reactive behaviours in the suggested control method.

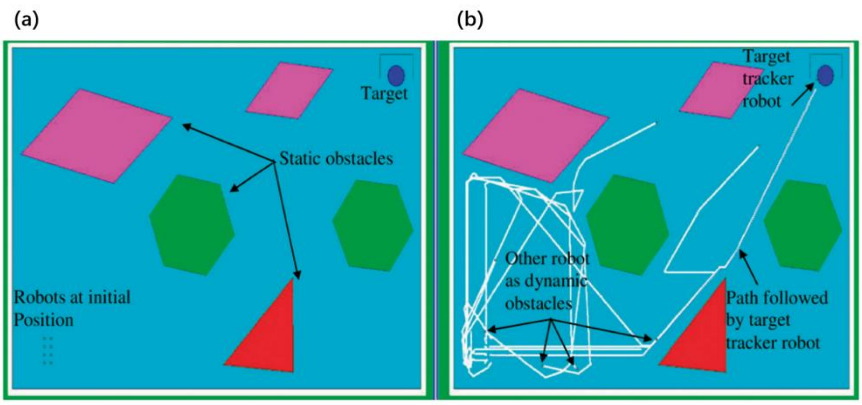

Figure 8. SVPPM generated path on a grid map [9].

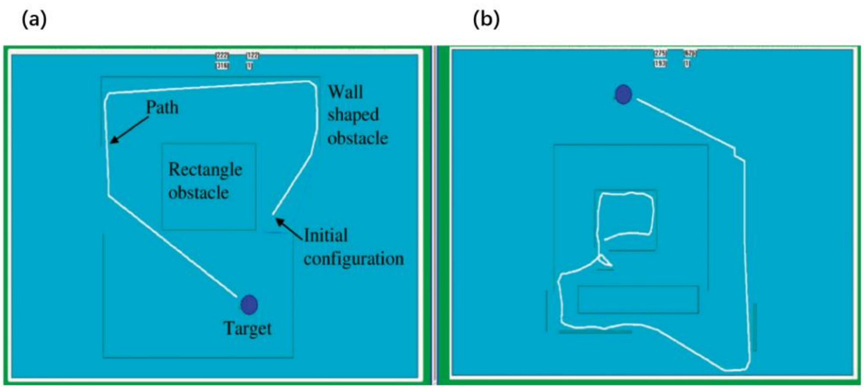

Figure 9. SVPPM generated path on a grid map [9].

Figure 10 shows the experimental pathways mobile robots used to approach the destination to test the control algorithms. The experimental trajectories closely matched those traced by the robots in Figure 11. It is caused by different errors. Different environmental situations were tested in the lab. Figures show that robots can avoid barriers and attain goals. Figure 11 shows the simulation and experimental verification of certain robotic behaviours. In both the simulation and the experiment, robots were able to achieve the objectives that they had established for themselves and were effective in doing so. The neural controller robot performed noticeably better than its fuzzy controller relative in terms of both the precision of its placement and its capacity to avoid collisions. This was the case regardless of the task at hand. Neural controllers need just a very small fraction of the amount of processing time and memory that fuzzy controllers do in order to reach the same level of performance as fuzzy controllers. This allows neural controllers to attain the same level of performance as fuzzy controllers.

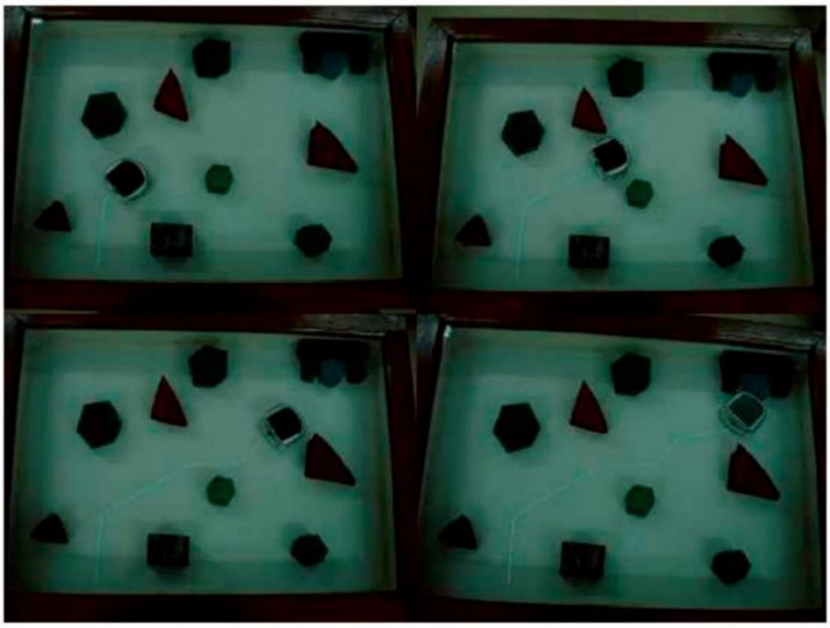

Figure 10. Experimental validation of navigation in the unknown environment [9].

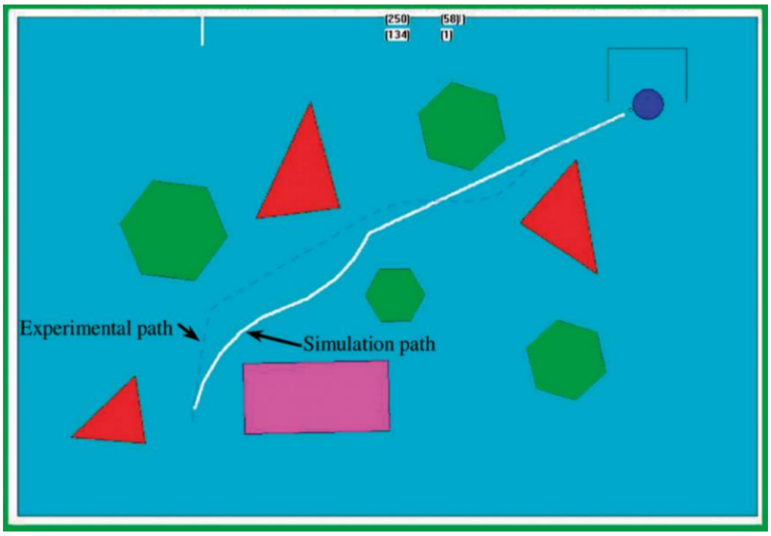

Figure 11. SVPPM generated path on a grid map [9].

DOI: 10.54254/2755-2721/5/20230639

4. Conclusion

From the methods mentioned above, the author finds several policies of Heuristic approaches. For the fuzzy logic technique, the path planning follows IF-THEN rules. By receiving the information, the fuzzy logic controller will output suitable motion to avoid obstacles and find its way. To make the robots get their destinations, positive rules and negative rules are combined to function. Also, the whole process will be clearer by dividing the fuzzy logic into 4 levels, as described in 'figure 1'. Level 1 is the lowest one, mainly to perceive the surrounding environment. Level 2 is mainly the monitoring and preprocessing. Level 3 is mainly the output of the commands. Level 4 is mainly the allocation of the motion signals. If the fuzzy logic is at a higher level, the robots' performance will be better, and the motion will look more robust. For neural networks in path planning, the neural network generally consists of four layers. They are the input layer, the output layer, and two hidden layers. The hidden layers are needed to be determined by the hidden nodes. Also, training the neural network is an important task. The training in [10] is divided into two phases. The first phase is offline training, and the second phase is online training. In online training, an evaluation function is used to judge the performance of the robot.

References

[1] Mac T, Copot C, Tran T and De R 2016 Heuristic approaches in robot path planning: A survey *Robotics and Autonomous Systems* 86 13-28

[2] Iswanto I, Ma'arif A, Wahyunggoro O and Cahyadi I 2019 Artificial potential field algorithm implementation for quadrotor path planning *International Journal of Advanced Computer Science and Applications* 10 575-585

[3] Primanita A and Hiroyuki I 2020 Nature of probability-based proof number search *Sriwijaya International Conference on Information Technology and Its Applications (SICONIAN 2019)* 172 485-491

[4] Zhang L and Manocha D 2008 An efficient retraction-based RRT planner *2008 IEEE International Conference on Robotics and Automation* 3743-3750

[5] Caihong L, Fengying W, Yong S, Zhenying L and Zhiqiang W 2015 A complete coverage path planning algorithm for mobile robot based on FSM and rolling window approach in unknown environment *2015 34th Chinese Control Conference* 5881-5885

[6] Fadzli A, Abdulkadir I, Makhtar M and Jamal A 2015 Robotic indoor path planning using dijkstra's algorithm with multi-layer dictionaries *2015 2nd International Conference on Information Science and Security* 1-4

[7] Abdessemed F, Faisal M, Emmadeddine M, Hedjar R, Al-Mutib K, Alsulaiman M and Mathkour H 2014 A hierarchical fuzzy control design for indoor mobile robot *International Journal of Advanced Robotic Systems* 11 33

[8] Chang H and Jin T 2013 Command fusion based fuzzy controller design for moving obstacle avoidance of mobile robot *Future information communication technology and applications* 905-913

[9] Singh MK and Parhi DR 2011 Path optimisation of a mobile robot using an artificial neural network controller *International Journal of Systems Science* 42 107-120

[10] Al-Sagban M and Dhaouadi R 2012 Neural-based navigation of a differential-drive mobile robot *2012 12th International Conference on Control Automation Robotics & Vision* 353-358

[11] Chen J and Dames P 2020 Collision-free distributed multi-target tracking using teams of mobile robots with localization uncertainty *IEEE/RSJ International Conference on Intelligent Robots and Systems* pp. 6968-6974

[12] Chen J and Dames P 2021 Distributed multi-target tracking for heterogeneous mobile sensing networks with limited field of views *IEEE International Conference on Robotics and Automation* pp. 9058-9064