IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Journal Européen des Systèmes Automatisés
Vol. 55, No. 3, June, 2022, pp. 339-348
Journal homepage: http://iieta.org/journals/jesa

Artificial Techniques Based on Neural Network and Fuzzy Logic Combination Approach for Avoiding Dynamic Obstacles

Zead Mohammed Yosif[1*], Basil Shukr Mahmood[2], Saad Zaghlul Saeed[1]

[1] Mechatronics Engineering Department, Collage of Engineering, University of Mosul, Mosul 41002, Iraq
[2] Computer Engineering Department, Collage of Engineering, University of Mosul, Mosul 41002, Iraq

Corresponding Author Email: zmyousif@uomosul.edu.iq

https://doi.org/10.18280/jesa.550306

Received: 5 May 2022
Accepted: 7 June 2022

Keywords:
dynamic obstacle avoidance, fuzzy logic, neural network, mobile robot navigation, path planning

ABSTRACT

The autonomous mobile robot must be capable of avoiding static and dynamic obstacles in the environment and navigating towards the target without any human effort. A valid low-cost path from start to goal is obtained by A* algorithm. Neural network used for Zone classification. The relative values between mobile robot and obstacle are used for classification which are distance, velocity, and angle. Zone1 is very dangerous while zone 5 is not dangerous. If the neural network classifies the obstacle as a dangerous obstacle and activates the controller. The fuzzy logic makes a decision as a reaction of mobile robot to prevent collision. There are three inputs to the fuzzy logic (relative velocity, relative distance, and relative angle) between mobile robot and obstacle. The outputs of fuzzy logic are velocity and steering angle of mobile robot. Static obstacles have been added to the environment in addition to dynamic obstacles to make the environment more complex. Three dangerous dynamic obstacles to the mobile robot are tested. While mobile robot is avoiding one obstacle, another obstacle enters critical zone and becomes dangerous to mobile robot. The mobile robot avoids the second obstacle while it is avoiding the first obstacle. Then the velocities of mobile robot and obstacles have been increased to prove that the proposed system can handle cases with faster velocities. The simulation results for the tested cases shows the capability of the proposed method for avoiding static and dynamic obstacles in fully known environment.

1. INTRODUCTION

Mobile robots these days have been incorporated into various applications such as agriculture, military, medicine, education, mining, space travel, entertainment, etc. The autonomous mobile robot is an artificially intelligent machine capable of understanding the environmental situation, self-path planning by avoiding static and dynamic obstacles, and responding quickly to any environmental state without any human effort [1, 2]. The goal of navigation is to find an optimal or suboptimal path from the starting location to the destination point while avoiding obstacles [3, 4].

Neural networks (NNs) are a collection of non-linear, multi-layered, parallel regression techniques that can be used for signal processing, forecasting, and grouping [5]. Khanisi et al. [6] introduced a neural controller with two inputs representing the velocities of the right and left wheel, and two outputs representing the Pulse Width Modulation (PWM) for each wheel. Li et al. [7] used a neural network data fusion strategy is presented to reduce the affection induced by inaccuracies in the environment or measurements in order to enhance the real-time performance and accuracy of localization for mobile robots in interior environments. Abagiu et al. [8] developed a deep neural network for object identification in a mobile robot. A mobile robot was created and built to work in hazardous environments.

Fuzzy logic is employed in scenarios involving a lot of ambiguity, complexity, and nonlinearity. Batti et al. [9] proposed a fuzzy logic controller to reach the target in fully static environment, where three distances as inputs were entered into the fuzzy controller. The outputs are the right and left wheel velocities. Singh et al. [10] introduced a Fuzzy logic with two inputs to reach target in static environment. The inputs are the angle between the robot orientation and the robot goal orientation (i.e., robot rotation angle), and the distance between the obstacle and the robot which were determined by the ultrasonic sensor. The outputs are the angular velocities of the mobile robot's two wheels. Liu et al. [11] proposed an obstacle avoidance system based on fuzzy logic. The robot selects between increasing speed, lowering speed, or stopping to wait for the obstacle to pass based on the obstacle's movement trend. The robot avoided the obstruction simply by modifying the speed ahead of time, without changing the planed path.

The neural network and fuzzy logic are integrated with specific methods to create a new technology structure known as neural-fuzzy technology. To complete the fuzzy inference function, it can be handled in parallel as a neural network, self-learning, and fuzzy information can be processed as fuzzy theory [12]. Buduanto et al. [13] compared two artificial intelligence algorithms used in the wall following autonomous mobile robot: Neural Network Backpropagation and Fuzzy Logic. The distance between the robot and the wall is the input. The two wheels' speeds are the output variables. In a wall-following robot, the study discovered that NN is faster than Fuzzy Logic. Tan Ai and Dadios [14] discussed a neural

network and fuzzy logic combination. A neural network is utilized to learn both goal seeking and obstacle avoidance using a dataset created by two different fuzzy logic navigation algorithms. Table 1 shows the comparisons between methods.

Table 1. Methods comparisons

	NN	Fuzzy logic	NN and Fuzzy combination
Path planning types	Dynamic and static path planning	Dynamic path planning	Dynamic and static path planning
Environment types	Dynamic and static	Dynamic and static	Dynamic and static
Training data	Yes	No	Depend on NN
Memory usage	High	Low	Depend on NN
Applications	Velocity controlling, path planning, object identifications	Controlling robot motors, suggest steering angle, path planning	Velocity controlling, path planning, suggest steering angle, obstacle avoidance.

From the above comparison, it is clear that fuzzy logic effective for motion control due to its fast response and low memory and calculations requirements. The neural network with its training provides classifications for different situations that might be happened.

Various mobile robot navigation techniques have been applied by the researchers to achieve collision free path planning. Many papers tried to make a combination between two algorithms or more to increase the efficiency. The reactive approach can handle the problem of dynamic environment more that static environment. This paper concerns with reactive algorithms to handle problem of dynamic obstacle avoidance by finding initial path using A* algorithm. Then a combination between neural network and fuzzy logic have been done to achieve obstacle avoidance. The neural network classifies the zone; while fuzzy logic controls the speed and orientation of the robot so as to avoid the static and dynamic obstacles for mobile robot travelling on indoor environment. This paper is organized as follows: In section 2 path planning is presented. Section 3 presents zone classification using neural network. In section 4, the fuzzy logic input – outputs and rules are introduced. Result and discussion have been presented in section 5. The conclusion and suggestion are given in section 6.

2. PATH PLANNING

The A-star algorithm, often known as A*, is a heuristic search method whose goal is to identify the shortest path in an environment from a start point to a goal point as shown in Figure 1. The A* algorithm is a modification of the Dijkstra algorithm. A-star is considered the most representative algorithm [15]. The task of determining a valid low-cost path from start to goal locations in an environmental map is referred to as path planning. The basic method starts with the shortest path and then expands it with the function indicated below [16].

$$f(n) = g(n) + h(n) \quad (1)$$

where,
g(n): the cost of getting path from the starting point to node (n).
h(n): denotes a heuristic estimate, cost, or path from node n to a target.

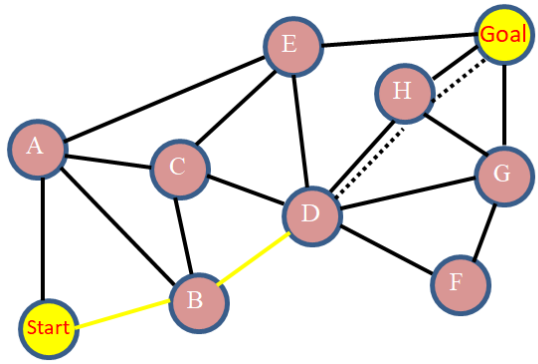


Figure 1. A* path planning

Search-based planning, which includes A* search algorithm, is a prominent solution to path planning issues that has been applied in a variety of applications including autonomous vehicle navigation, robot arm manipulation, and gaming AI [17]. If the environment has previously been completely mapped, A* can be employed in almost every scenario [18]. The A* search Algorithm can be used to find the shortest route to a robot's final point [19, 20].

3. ZONE CLASSIFICATION

In depending on distance between mobile robot and obstacle, relative velocity between mobile robot and obstacle, and angle between mobile robot and obstacle, this work proposes five zones. Zone 1 is very dangerous while zone 5 is not dangerous. If the obstacle is close to the robot, it will be considered as a dangerous. Also if the relative velocity is high, the obstacle can be classified as dangerous. On other hand; if the obstacle is far from mobile robot, it can be classified as not dangerous. Table 2 shows zone's classifications depending on self-experience. The velocity and distance between the obstacle and the mobile robot are more important than the angle between them. Velocity and distance pose the greatest danger to the robot and to the possibility of collision. High speed and the closest distance between the mobile robot and the obstacle increase the possibility of collision. Thus, these two parameters are included in the Table 2, and that each field of the table includes angles from 90 to - 90 degrees.

Table 2. Zone's classifications

Distance (cm) / Velocity (cm/s)	30	60	90	120 and more
4	Zone3	Zone4	Zone5	Zone5
6	Zone2	Zone3	Zone4	Zone5
8	Zone1	Zone2	Zone3	Zone4
10	Zone1	Zone1	Zone3	Zone3

As an example, obstacle 1 (Obs1) has velocity = 5 cm/sec, distance = 120 cm, and angle = 0o. Obstacle 2 (Obs2) has velocity = 5 cm/sec, distance = 60 cm, and angle = 0o. These information will be arranged in the following form [distance,

velocity, angle] input vector. for Obs1 and Obs2 will be as flowing form [120,15,0], [80,15,0]; respectively as shown in Figure 2. Obstacle 1 and obstacle 2 have same velocity and angle but the distance between mobile robot and obstacles are different. At the output side there is a vector of five elements. The output is five coded number in which each digit may be 0 or 1 because there are five suggested zones. The suggested zone equals to 1. For example, if the vector is [0,1,0,0,0] this means that zone two is activated. Obstacle 2 is more dangerous than obstacle 1. Therefore; it should have zone classification highest than obstacle 1.

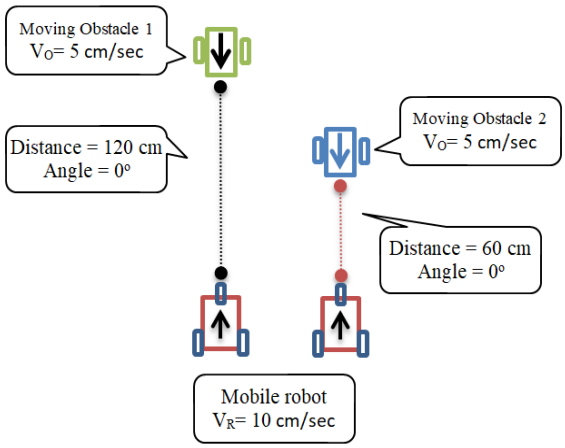
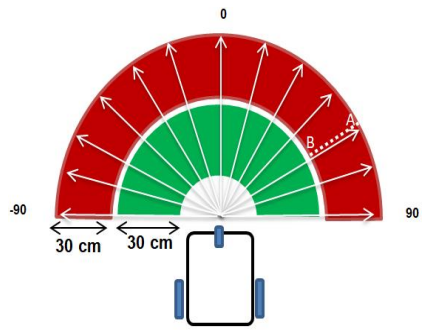
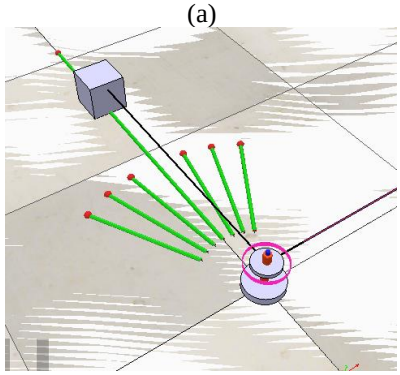


Figure 2. Robot's zones example

3.1 Data collection



(a)



(b)

Figure 3. Spectrum of path's angle

For each zone of velocity and distance seven angles are included that start from 0o- 90o degrees that are located in the right side of robot, and also 0o to – 90odegrees that are located in the left side of robot as it clear in Figure 3. For zones in Table 1, ten readings are recorded along the radial path from obstacle to mobile robot. Also for zone 1 where the velocity= 10 cm/sec, and distance = 60 cm it needs to cover a degrees (-90)-(-75)-(-60)-(-45)-(-30)-(-15)-0-15-30-45-60-75-90. Therefore; a 130 readings are obtained. The total readings will be 16·130= 2080. The radial reading for each angle will start from distance 60 cm until 30 cm. Matlab and robot simulator CoppeliaSim are used to record these readings. Reading steps and resolution are represented in dotted line in the red color that cover form -90o to 90o as shown in Figure 3.

3.2 CoppeliaSim and data collection

The Virtual Robot Experimentation Platform (V-REP) (now called CoppeliaSim) is the premier robotics and simulation automation software platform in the world. Through a rich application programming interface (API), V-REP allows the user to model the entire robotic system or its subsystems (such as sensors or actuators), as well as simply integrate the robot's operations. Each simulated object or model can be individually controlled using embedded scripts or a remote API client using the V-REP integrated development environment and distributed control architecture [21]. It has the ability to add objects and sensors. In order to collect data, a distance senor has been mounted on the rotary desk. An object with specified velocity toward the sensor is sensed. The moving object is moved in directly forward line toward the sensor with certain angle. The angle distribution is shown in Figure 3. The zero angle is in the centre. The right side range goes to positive angle until reach +90 degree, while the left side range goes to negative angle until reach to – 90 degree. For certain angle there are ten reading have been acquired. The reading resolution is as referred between B and A points. The distance between each point is 3 cm in a total distance of 30 cm between two areas. The data used in the research was obtained and generated based on personal experiences and self-experience.

3.3 Neural network

An ANN is a bio-inspired artificial brain model capable of mimicking behavior-based learning. The basic computing unit of an ANN is a neuron, which has the ability to store and replicate experiential information in a manner comparable to the human brain. These have been widely employed in numerous search optimization, learning, and pattern recognition problems due to their ability to produce simple and optimal solutions in complex scenarios while maintaining the integrity of the specifications [22]. A feedforward net is one that was trained using the backpropagation training algorithm. To obtain the error signal, the backpropagation training algorithm subtracts the training output from the goal (desired response). The weights and biases in the input and hidden layers are then adjusted to minimize the error [23]. A feed forward neural network trained to be used in zone classification. There are three inputs (velocity, distance and angle), ten neurons in the hidden layer, and five neurons in the output (five zones) as it shown in Table 3.

Table 3. Zone output codes

Zone name	Neural output
Zone 1	00001
Zone 2	00010
Zone 3	00100
Zone 4	01000
Zone 5	10000

The mathematical equations are listed as below:

$$Ew = \frac{1}{2}\sum_{k=0}^{k}(t_k - O_k)^2 \quad (2)$$

The hidden layer's weights are updated using the equation:

$$Vji\,(n + 1) = Vji\,(n) + \eta \times \delta j\,(n) \times xi\,(n) \quad (3)$$

δj is the error signal produced by the j th hidden neuron.
The weights of the output layer are updated using the equation:

$$Wk\ j\ (n + 1) = Wk\ j\ (n) + \eta \times \delta k\ (n) \times y\ j\ (n) \quad (4)$$

δk is the error signal produced by the kth output neuron.

$$\delta k\,(n + 1) = (tk - Ok) \times (1 - Ok) \times Ok \quad (5)$$

Using δk, we can calculate δj as follows:

$$\delta j = (1 - y\,j) \times y\,j \times \Sigma k \quad (6)$$

where,
xi: The ith input.
y j: The output of the j th hidden neuron.
Ok: The output of the kth output neuron.
tk: The desired output.
Vji: The weight from the i^{th} input to the j^{th} hidden neuron.
Wk j: The weight from the i^{th} hidden neuron to the k^{th} output neuron.
η: The learning rate.
i: index for input neurons.

In order to train and test the network, a data base is considered for such that about 70% of the data for training, 15% of the data are used for validation and 15% are used for testing using 1000 epochs. The training accuracy was found about 93.5% and the validation accuracy at one-layer neural network is 92.6%. In a neural network, the number of neurons and hidden layers to be correctly guessed is determined by training database samples [24]. Therefore; the number of hidden layers is increased to a two hidden layers instead one hidden layer as shown in Figure 4. The accuracy is increased to 98% and validation accuracy is 97.2 as it is clear from Table 4. The confusion matrices for the two networks are shown in Figure 5.

Table 4. Specification of the used network

	Characteristics	Neural network 1	Neural network 2
1	**Number of input layer neurons**	3	3
2	**Number of hidden layer neurons**	10	10 , 10
3	**Number of output layer neurons**	5	5
5	**Hidden layer activation function**	Bayesian Regularization	Bayesian Regularization
6	**Output layer activation function**	Linear	Linear
7	**Learning rate**	0.05	0.05
8	**Maximum number of epoch**	1000	1000
	Accuracy	93.5%	98%
	Validation accuracy	92.6%	97.2%

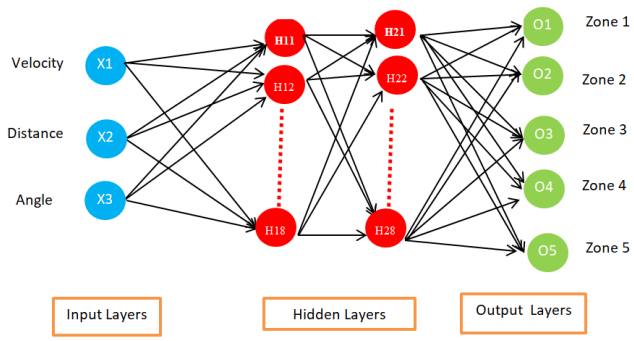
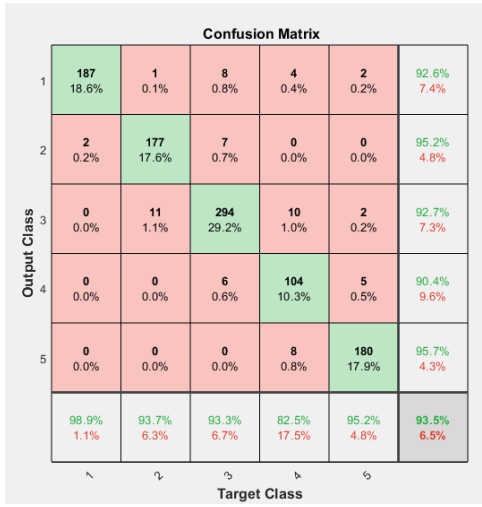
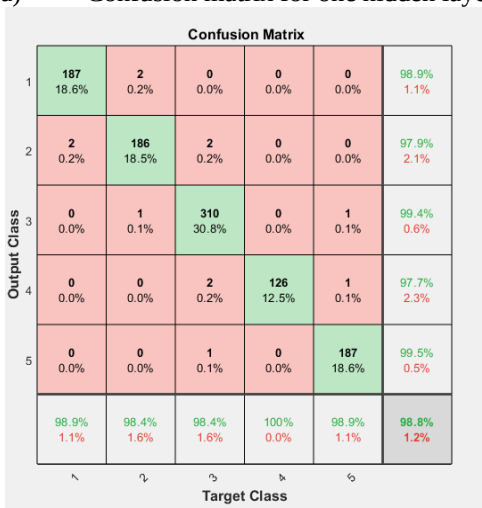


Figure 4. The structure of the used NN



(a) Confusion matrix for one hidden layer



(b) Confusion matrix for two hidden layer

Figure 5. Confusion matrix for the two networks

4. FUZZY LOGIC

A fuzzy system is a collection of fuzzy expert knowledge that can reason about facts in general terms rather than using strict Boolean logic. It was first established by Lotfi Zadeh in the 1960s, has become a prominent technique for control applications. A basic fuzzy system has four key components: a fuzzifier, a knowledge base, an inference engine, and a defuzzifier. The fuzzifier converts a real crisp input into a fuzzy function, identifying the input's 'degree of membership' to a vague idea. The controller's decision-making logic is provided by the Inference Engine. It uses fuzzy implications

and fuzzy inference procedures to deduce the fuzzy control actions. Defuzzification turns fuzzy control values into crisp numbers, i.e., it connects a single point to a fuzzy set if the point is part of the fuzzy set's support [25].

There are three inputs to the fuzzy logic system which are (relative velocity, relative distance, and relative angle) between the mobile robot and the obstacle. The output of the fuzzy logic system (in other word the decision of fuzzy logic) is velocity and steering angle of the mobile robot as shown in Figure 6. When mobile robot enters a dangerous zone that is classified by NN, the fuzzy system decides if the velocity of the robot must be changed or not and if the robot steering angle will be changed or not.

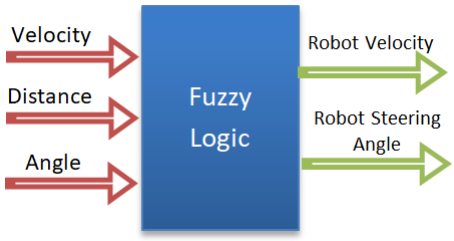


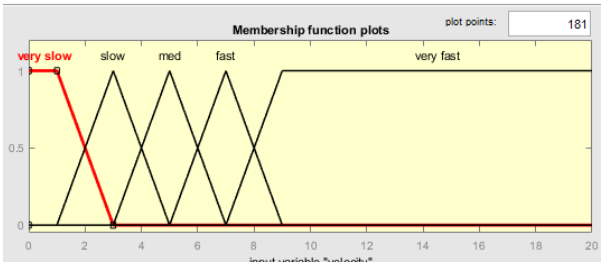
Figure 6. Suggested fuzzy logic system

Each fuzzy input has five memberships. The velocity memberships are considered as: very slow, slow, med, fast, very fast. The distance memberships are: very near, near, med, far, very far. The steering angle memberships are: sharp left, left, forward, right, sharp right. Each fuzzy output has five memberships. The steering angle memberships are: left, fine left, forward, fine right, and right. The velocity memberships are: very slow, slow, med, fast, very fast. The memberships of fuzzy inputs and outputs shown are shown in Figure 7. There are five memberships for each input, Therefore the number of the required rules become 5*5*5= 125 rules that control, the fuzzy system. Figure 8 displays part of the rules viewer. Figure 8, a sample of the 125 rules that were written in fuzzy logic, where Figure 8-a shows the IF-THEN rules that used in the controller which consists of three inputs: relative velocity, distance, and angle between the robot and the obstacle, while at the output end are: the speed and the angle of rotation of the robot so that it can avoid collision. The rules are constructed based on following pattern:

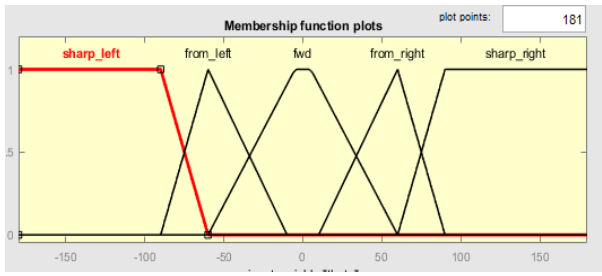
Rule i: If velocity is (Vi) and distance is (Di) and angle is (Ai) then the velocity is Vo and steering angle is Ao.

Where as Vi, Di, and Ai the input parameters. Vo and Ao output parameters.

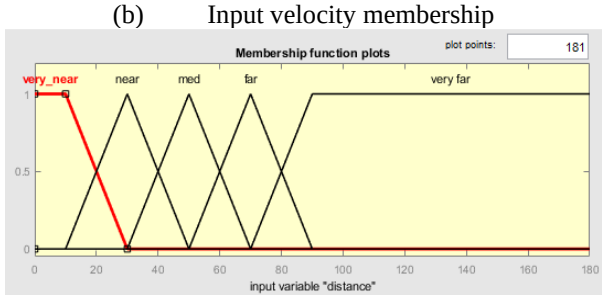
In Figure 8-b shows the rule-view and the memberships graphs that formed for all IF-THEN rules based on the input parameters of fuzzy system.



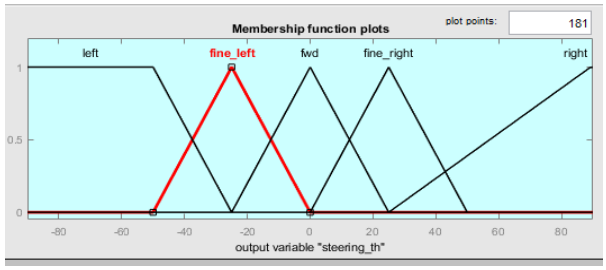
(a) Input velocity membership



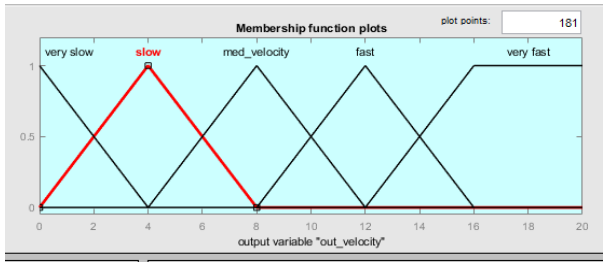
(b) Input velocity membership



(c) Input distance membership

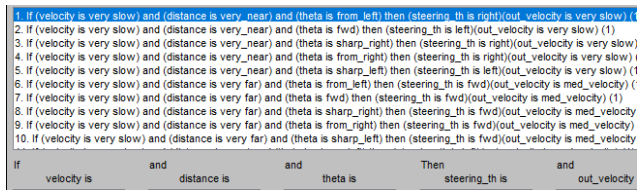


(d) Output velocity

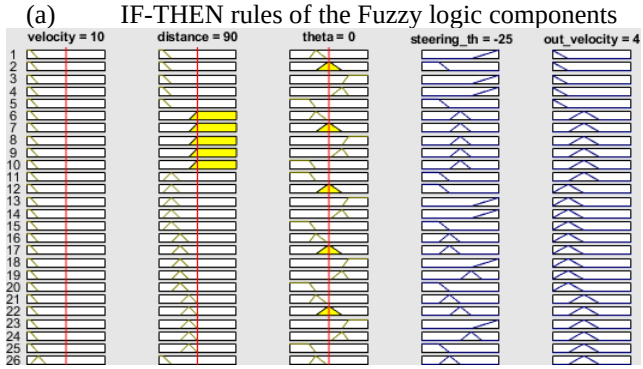


(e) Output angle

Figure 7. Fuzzy logic memberships



(a) IF-THEN rules of the Fuzzy logic components



(b) Rule – view of the Fuzzy logic component

Figure 8. Fuzzy logic rules viewer

5. RESULTS AND DISCUSSION

An environment with 500 * 500 cm^2 is suggested in order to test the NN and fuzzy logic system that have been designed. The start point is (200, 100) and the goal point is (200, 400). As the mobile robot enters a dangerous region, the neural network classifies the obstacle as a dangerous obstacle which may collide the robot. After this classification the fuzzy logic makes a decision as a reaction of mobile robot to prevent collision. The proposed algorithm is presented in Figure 9.

At first (test 1), a free environment with only one dynamic obstacle has been tested. The obstacle is moving in front of the mobile robot with heading angle equals to zero degree. As the obstacle gets into dangerous zone, the mobile robot velocity and steering angle will be changed according to the fuzzy rules. Another case (Test 2) has been tested that it is similar to the first case but both mobile robot and obstacle are moving faster than that in the first case. At (Test 3 and Test 4) the obstacle moves in front of mobile robot but with angle about 45 degree to the right and to the left of the mobile robot; respectively. For all previous cases (Test 1-Test 4) zones are classified as dangerous zones. For the next two testes (Test 5 & Test 6) the obstacle is moving in front of the mobile robot but it is parallel to robot. The obstacle is classified as not dangerous to the robot. Table 5 shows the initial path length that have been found by A-star algorithm, the path after avoiding obstacles, velocity of both obstacle and mobile robot, and the relative velocity. Figure 10 shows the tested cases.

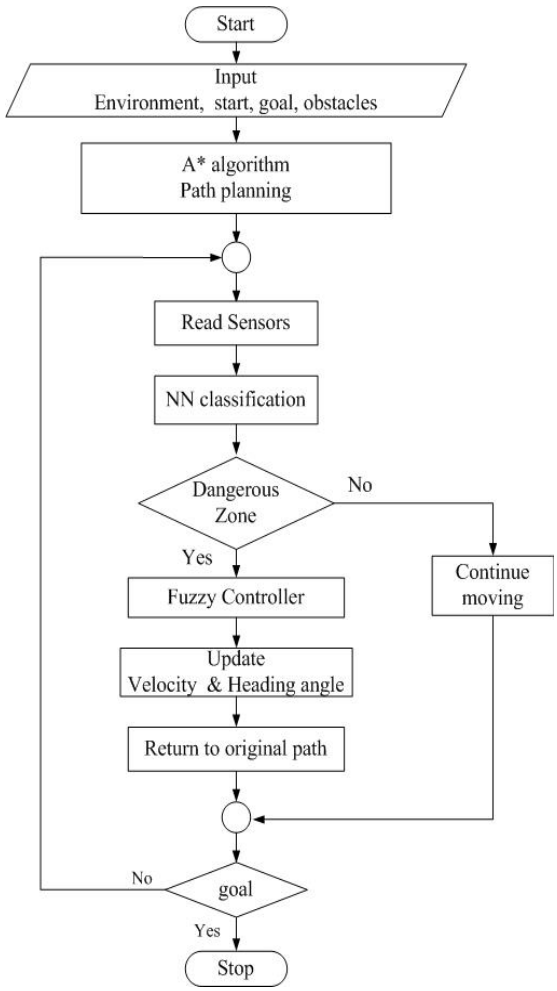
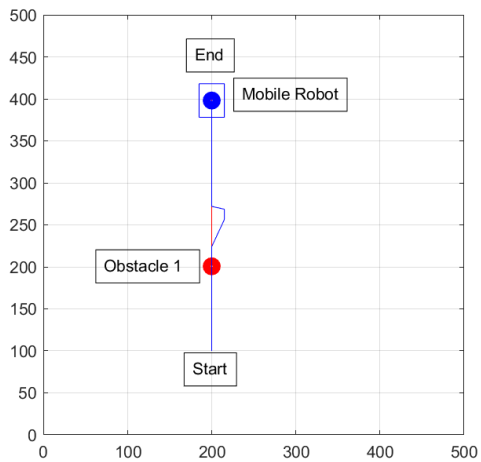


Figure 9. Algorithm flowchart for proposed neural fuzzy combination

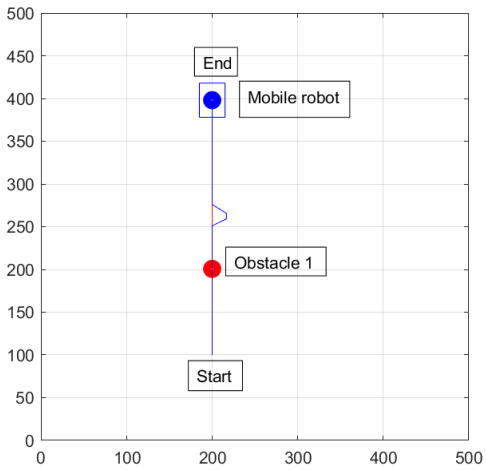
Table 5. Results and tests (Test 1 – Test 6)

	Test1	Test2	Test3	Test4	Test5	Test6
Initial path length using A*	300	300	300	300	300 cm	300
Time required for initial path (sec)	0.651	0.651	0.651	0.651	0.681	0.681
direct length from start to end points	300	300	300	300	300 cm	300 cm
Path length after avoidance (cm)	315	316	319	319	300 cm Parallel to robot	300 parallel to robot
Obstacle velocity (cm/s)	10	15	10	10	10	10
Robot velocity (cm /s)	10	15	11	11	10	10
Relative velocity (cm /s)	20	30	19	19	20	20
Notes on path	Direct	Direct	Form left	From right	No collision – parallel	No collision
Increased path length ratio %	5	5.3	6.3	6.3	0	0

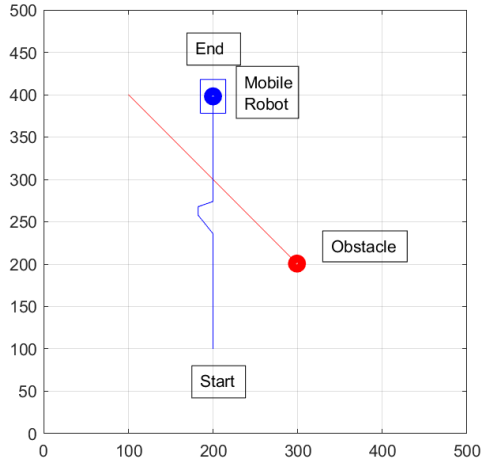
In the second scenario of tests another obstacle has been added to the environment. Thus two obstacles may collide the robot. In test 07, obstacle 1 is moving straight forward to the mobile robot which may collide the mobile robot. The mobile robot must avoid it. The Obstacle 2 is moving in front of mobile robot but has no chance of colliding the mobile robot. In Test 8 both obstacles move in front of mobile robot but in parallel to the robot path therefore the mobile robot does not change initial path. In Test 9 and Test 10, one obstacle is chosen to be dangerous and the other is not. For this reason, the mobile robot avoids the obstacle and returns to its initial path. In test 11 both obstacles enter critical zone. The mobile robot avoids the first obstacle and returns to its initial path. After that, it then avoids the second obstacle and returns to the initial path again. The results of the tested cases are presented in Table 6 and Figure 11.



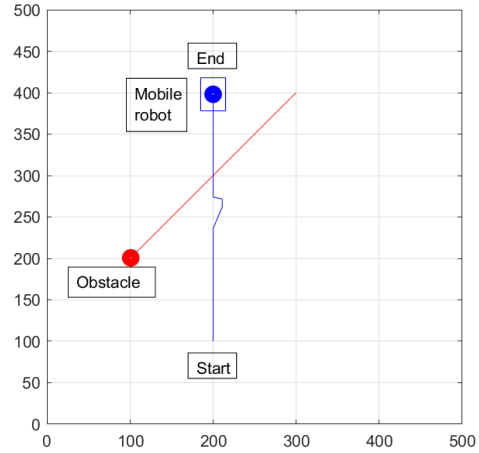
Test 01



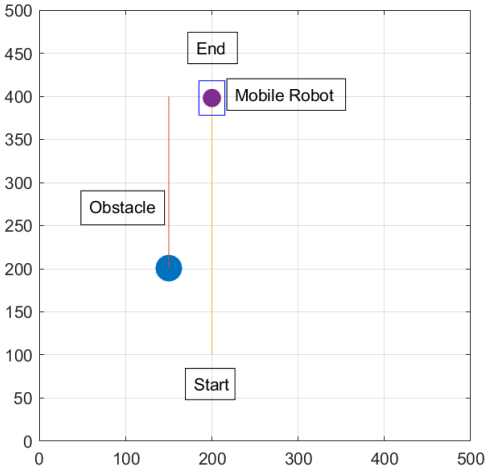
Teas 02



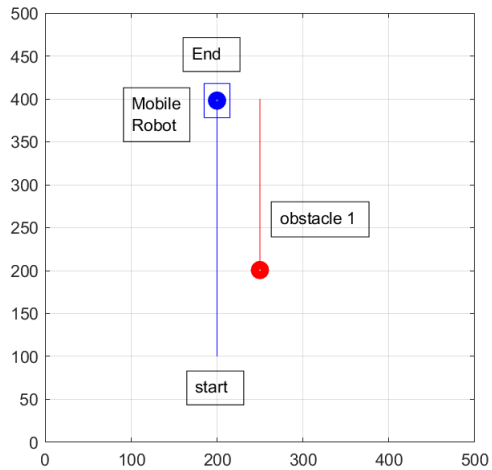
Test 03



Test 04

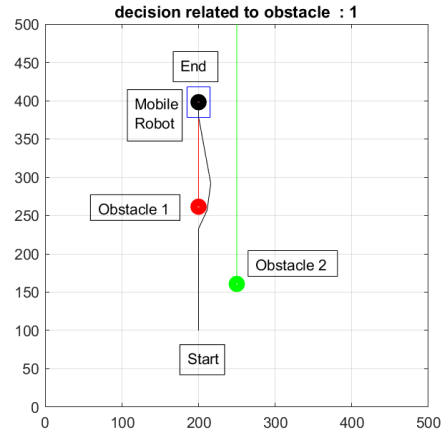


Test 05

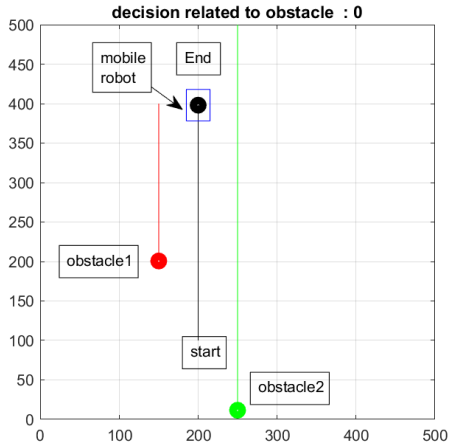


Test 06

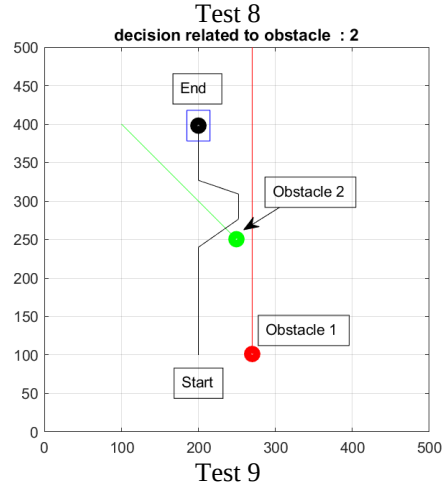
Figure 10. Test scenario for one moving obstacle



Test 7



Test 8



Test 9

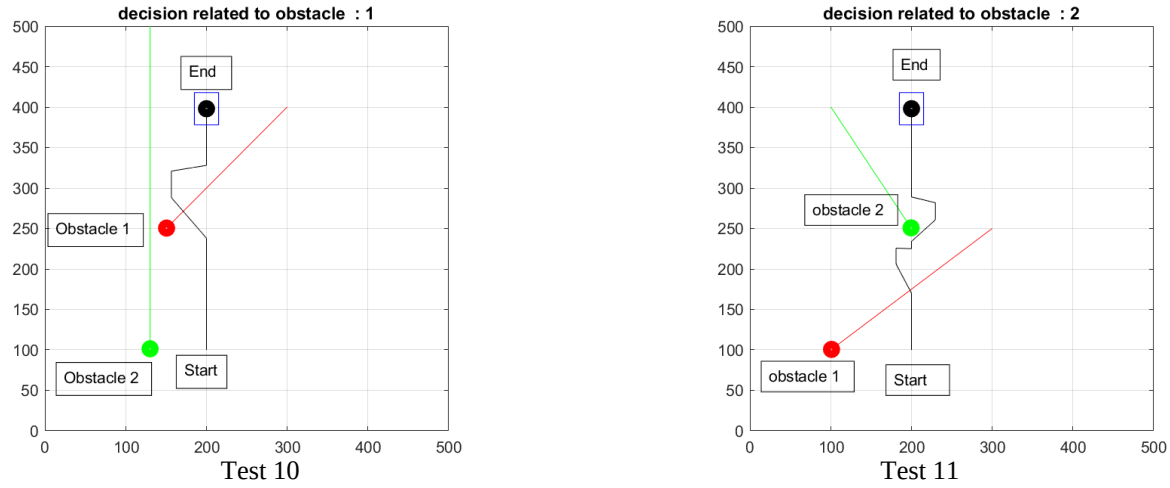


Figure 11. Scenario test for two moving obstacles

Table 6. Results and tests (Test7-Test 11)

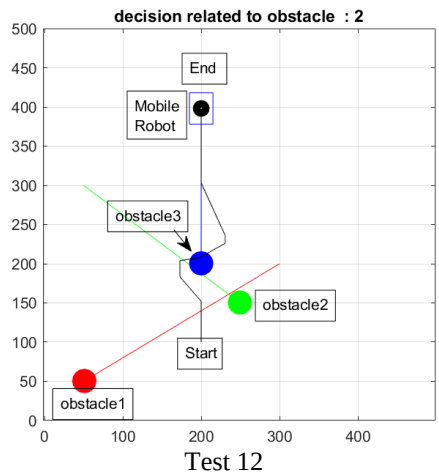
	Test7	Test8	Test9	Test10	Test11
Initial path length using A*	300	300	300	300	300
Time required for initial path (sec)	0.681	0.651	0.651	0.651	0.651
direct length from start to end points	300 cm	300	300	300	300
Path length after avoidance (cm)	360	353	364.4322	300	304
Obstacle 1 velocity (cm/s)	11	11	11	10	6
Obstacle 2 velocity (cm/s)	10	6	6	14	14
Robot velocity (cm /s)	11	14	10	10	8
Relative velocity 1 - (cm /s)	20	21	21	20	12
Relative velocity 2 - (cm /s)	20		14	24	20
Increased path length ratio %	20	17.7	21.3	0	1.3

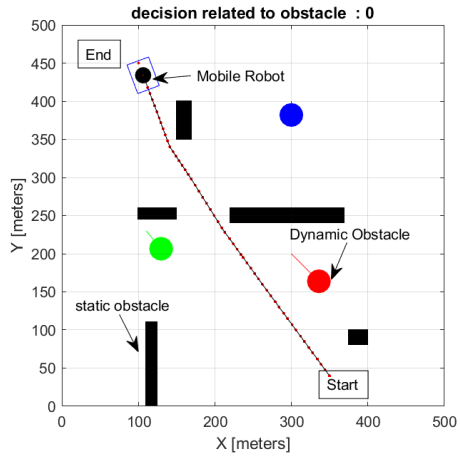
Table 7. Results and tests (test 12- test 17)

	Test12	Test13	Test14	Test15	Test16	Test17
Initial path length using A*	540	546	541	483	541	300
Time required for initial path (sec)	9	17	12	3	10	0.707
direct length from start to end points	480	480	480	480	480	300
Path length after avoidance (cm)	652	666	514	483	541	358
Obstacle 1 velocity (cm/s)	35	20	10	10	10	9
Obstacle 2 velocity (cm/s)	40	20	10	10	10	8
Obstacle 3 velocity (cm/s)	40	20	10	10	10	5
Robot velocity (cm /s)	80	42	70	73	70	10
Relative velocity 1 - (cm /s)	115	61	62	70	62	15
Relative velocity 2 - (cm /s)	71	52	62	67	62	15
Relative velocity 3 - (cm /s)	105	61	62	65	62	14
Increased path length ratio %	34.90%	37.80%	6.40%	0	0	19.30%

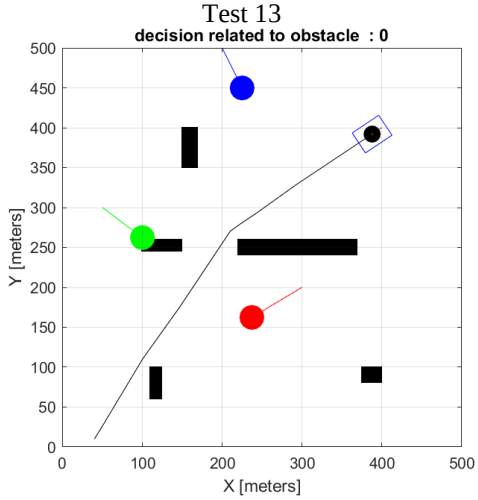
In the third scenario, there are three dynamic obstacles and all of them are dangerous to the mobile robot. In Test 12, while mobile robot is avoiding obstacle 2, obstacle 3 enters critical zone and becomes dangerous to mobile robot. The mobile robot avoids obstacle 3 while it is avoiding obstacle 2.

In the fourth scenario, static obstacles have been added to the environment in addition to dynamic obstacles to make the environment more complex. Test 13 & Test 14 are examples of moving mobile robot form start position to goal in presence of static and dynamic obstacle. But in these cases, the dynamic obstacles are not dangerous to the mobile robot. Thus it remains moving along the initial path. In test 15, two obstacles (obstacle 2 &3) are dangerous to the mobile robot. The mobile robot avoids these obstacles safely. Obstacle 1 exists in the environment but it is not dangerous to the mobile robot. In tests 16 and 17, three dynamic obstacles are dangerous to the mobile robot as it is clear in Table 7 and Figure 12. The mobile robot avoids them. In test 17 the velocities of mobile robot and obstacles have been increased to prove that the proposed system can handle cases with faster velocities.

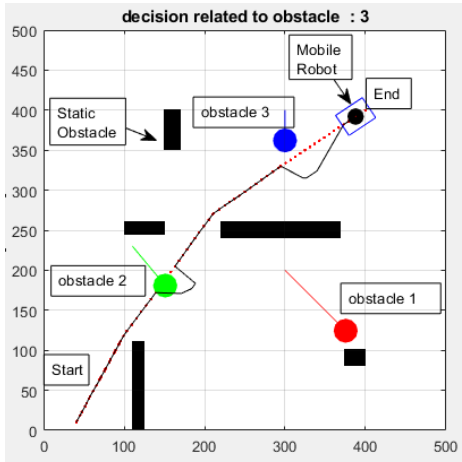




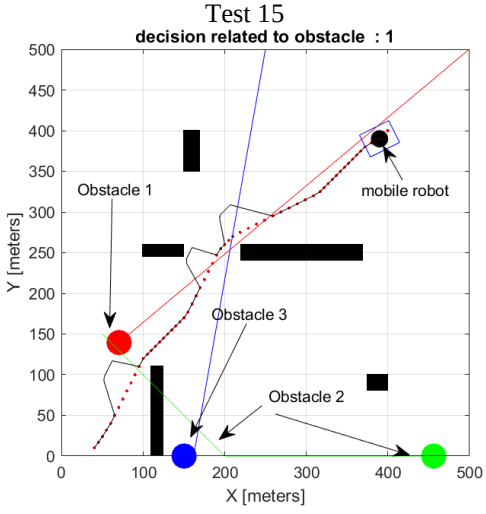
Test 13



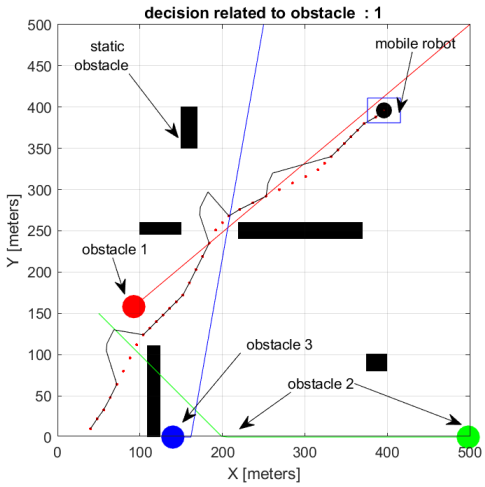
Test 14



Test 15



Test 16



Test 17

Figure 12. Scenario tests for three moving obstacles and static obstacles

6. CONCLUSION

For autonomous mobile robot applications, there are several situations to be handled due to the presence of static and dynamic obstacles. The data generated and collected by the simulator had provided the necessary set to train the neural network so as to achieve zone's classification. Obstacles are classified into five zones by using the two hidden layers neural network. In dangerous zone, fuzzy logic controls the mobile robot navigation through the changing of velocity and steering angle of mobile robot. The five memberships for each fuzzy controller inputs and outputs are sufficient to produce the reaction so as to avoid dynamic obstacles and continue navigation towards the goal. Complex situations are tested in which more than one dynamic obstacles at the same time are avoided even when the velocities of mobile robot and obstacles are doubled. The combination between neural network classification and fuzzy controller is succeeded to navigate the mobile robot in seventeen tests avoiding static and dynamic obstacles in fully known environment. The proposed model succeeded in determining which of the obstacles in the environment is the most dangerous for the robot when we have more than one moving obstacle at the same time. The robot also has the ability of continues checking the obstacles even during the implementation of the avoidance subprogram. In the future we propose a fusion between the proposed system model with prediction strategies to achieve better performance.

ACKNOWLEDGMENT

The authors would like to express their gratitude to the faculty and staff of the University of Mosul-College of Engineering, especially the departments of computer engineering and mechatronics engineering, for their support to complete this work.

REFERENCES

[1] Siegwart, R., Nourbakhsh, I.R., Scaramuzza, D. (2011). Introduction to Autonomous Mobile Robots. MIT Press.

[2] Patle, B.K. (2016). Intelligent navigational strategies for multiple wheeled mobile robots using artificial hybrid methodologies. National Institute of Technology Rourkela.
[3] Pandey, A., Pandey, S., Parhi, D.R. (2017). Mobile robot navigation and obstacle avoidance techniques: A review. Int Rob Auto J, 2(3): 00022. https://doi.org/10.15406/iratj.2017.02.00023
[4] Hutabarat, D., Rivai, M., Purwanto, D., Hutomo, H. (2019). Lidar-based obstacle avoidance for the autonomous mobile robot. 2019 12th International Conference on Information Communication Technology and System (ICTS), pp. 197-202. https://doi.org/10.1109/ICTS.2019.8850952
[5] Shanmuganathan, S. (2016). Artificial neural network modelling: An introduction. Artificial Neural Network Modelling, Springer, 2016, pp. 1-14. https://doi.org/10.1007/978-3-319-28495-8_1
[6] Khnissi, K., Seddik, C., Seddik, H. (2018). Smart navigation of mobile robot using neural network controller. in 2018 International Conference on Smart Communications in Network Technologies (SaCoNeT), pp. 205-210. https://doi.org/10.1109/SaCoNeT.2018.8585616
[7] Li, H., Mao, Y., You, W., Ye, B., Zhou, X. (2020). A neural network approach to indoor mobile robot localization. 2020 19th International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES), pp. 66-69. https://doi.org/10.1109/DCABES50732.2020.00026
[8] Abagiu, M., Popescu, D., Manta, F.L., Popescu, L.C. (2020). Use of a deep neural network for object detection in a mobile robot application. 2020 International Conference and Exposition on Electrical and Power Engineering (EPE), pp. 221-225. https://doi.org/10.1109/EPE50722.2020.9305648
[9] Batti, H., Jabeur, C.B., Seddik, H. (2019). Mobile robot obstacle avoidance in labyrinth environment using fuzzy logic approach. 2019 International Conference on Control, Automation and Diagnosis (ICCAD), pp. 1-5. https://doi.org/10.1109/ICCAD46983.2019.9037873
[10] Singh, N.H., Thongam, K. (2018). Mobile robot navigation using fuzzy logic in static environments. Procedia Computer Science, 125: 11-17. https://doi.org/10.1016/j.procs.2017.12.004
[11] Liu, Y., Chen, D., Zhang, S. (2018). Obstacle avoidance method based on the movement trend of dynamic obstacles. 2018 3rd International Conference on Control and Robotics Engineering (ICCRE), Nagoya, pp. 4550. https://doi.org/10.1109/ICCRE.2018.8376431
[12] Wang, H., Duan, J., Wang, M., Zhao, J., Dong, Z. (2018). Research on robot path planning based on fuzzy neural network algorithm. 2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), pp. 1800-1803. https://doi.org/10.1109/IAEAC.2018.8577599
[13] Budianto, A., Pangabidin, R., Syai'in, M., et al. (2017). Analysis of artificial intelligence application using back propagation neural network and fuzzy logic controller on wall-following autonomous mobile robot. 2017 International Symposium on Electronics and Smart Devices (ISESD), pp. 62-66. https://doi.org/10.1109/ISESD.2017.8253306
[14] Tan, R.J.C., Dadios, E.P. (2018). Neuro-Fuzzy mobile robot navigation. 2018 IEEE 10th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM), pp. 1-6. https://doi.org/10.1109/HNICEM.2018.8666348
[15] Tang, Z.Z., Ma, H.Z. (2021). An overview of path planning algorithms. OP Conf. Ser.: Earth Environ. Sci., 804: 022024. https://doi.org/10.1088/1755-1315/804/2/022024
[16] Karur, K., Sharma, N., Dharmatti, C., Siegel, J.E. (2021). A survey of path planning algorithms for mobile robots. Vehicles, 3(3): 448-468. https://doi.org/10.3390/vehicles3030027
[17] Yonetani, R., Taniai, T., Barekatain, M., Nishimura, M., Kanezaki, A. (2021). Path planning using neural A* search. Proceedings of the 38th International Conference on Machine Learning, pp. 12029-12039. https://doi.org/10.48550/arXiv.2009.07476
[18] GopiKrishnan, S., Shravan, B.V.S., Gole, H., Barve, P., Ravikumar, L. (2011). Path Planning Algorithms: A comparative study. Space Transportation Systems.
[19] Aprilia, B.S., Kurniawan, E., Ramdhani, M., Rizal, A. (2019). Design and implementation A* algorithm on movement system robot in the warehouse. J. Phys.: Conf. Ser., 1367(1): 012066. https://doi.org/10.1088/1742-6596/1367/1/012066
[20] Wang, X., Mizukami, Y., Tada, M., Matsuno, F. (2021). Navigation of a mobile robot in a dynamic environment using a point cloud map. Artif Life Robotics, 26(1): 10-20. https://doi.org/10.1007/s10015-020-00617-3
[21] Tursynbek, I., Shintemirov, A. (2020). Modeling and simulation of spherical parallel manipulators in CoppeliaSim (V-REP) robot simulator software. 2020 International Conference Nonlinearity, Information and Robotics (NIR), pp. 1-6. https://doi.org/10.1109/NIR50484.2020.9290227
[22] Abed, M.S., Lutfy, O.F., Al-Doori, Q. (2021). A review on path planning algorithms for mobile robots. Engineering and Technology Journal, 39(5A): 804-820. http://dx.doi.org/10.30684/etj.v39i5A.1941
[23] Pwasong, A., Sathasivam, S. (2016). A new hybrid quadratic regression and cascade forward backpropagation neural network. Neurocomputing, 182: 197-209. http://dx.doi.org/10.1016/j.neucom.2015.12.034
[24] Uzair, M., Jamil, N. (2020). Effects of hidden layers on the efficiency of neural networks. 2020 IEEE 23rd International Multitopic Conference (INMIC), pp. 1-6. http://dx.doi.org/10.1109/INMIC50486.2020.9318195
[25] De Silva, C.W. (2018). Intelligent Control: Fuzzy Logic Applications. CRC Press, 2018.