IUST
Iran University of Science and Technology

Iranian Journal of Electrical and Electronic Engineering

Journal Homepage: ijeee.iust.ac.ir

A Combination of Adaptive Neuro-Fuzzy Inference System and Neural Network for Mobile Robot Dynamic Obstacle Avoidance

Zead Mohammed Yosif *(C.A.), Basil Shukr Mahmood**, Saad Z. Alkhayat*, and Aws Hazim Saber Anaz*

Abstract: A mobile robot must be autonomous to avoid obstacles while traveling towards the target. Dynamic obstacle avoidance remains a significant challenge in mobile robotics. Although reactive navigation strategies have been applied to address this problem, relying on the single-stage module often results in limited efficiency and restricted overall performance. This paper proposes combining an adaptive neuro-fuzzy inference system (ANFIS) and a neural network (NN). The data for obstacle severity classification were used to train the Neural Network. The relative velocity and distance between the mobile robot and obstacles determine the zone. Zone 1 is dangerous, and Zone 5 is safe. This paper uses the ANFIS to avoid obstacles during the mobile robot's motion and to avoid collisions. Based on our empirical study, three essential features have been considered in this paper: the relative speed, distance, and angle between the robot and the obstacle as inputs to the obstacle avoidance system ANFIS. The output was a suggested steering angle and speed for the mobile robot. The simulation results for the tested cases show the capability of the proposed controller to avoid static and dynamic obstacles in a fully known environment. Our results show that the ANFIS System enhances the proposed controller's performance, reducing path length, processing time, and the number of iterations compared to state-of-the-art research papers. The proposed work demonstrated better performance in path length reduction (approximately 6%) and time taken reduction to reach the target, which is reduced by about 60%.

Keywords: Mobile Robot Navigation, Dynamic Obstacle Avoidance, Neural Network, Adaptive Neuro-Fuzzy Inference System, Path Planning.

1. Introduction

THE autonomous mobile robot can be defined as a machine combined with artificial intelligence techniques to understand the surrounding environment conditions and find a path in the presence of dynamic and static obstacles [1]. The autonomous mobile robot should be able to respond to environmental conditions without human aid. The navigation strategies can be classified into local and global navigation depending on prior information about the environment. Global navigation concerns a completely known environment, while local navigation is related to unknown or partially known environments[2]. Perception, localization/ mapping, path planning, and motion control are the main parts of mobile robot navigation. The sensor-based data collection process is the perception step. The gathered information is utilized to create a map of the area (mapping). These sensors are also employed in localization, establishing the robot's position. Path

Iranian Journal of Electrical & Electronic Engineering, 2026.
Paper first received 24 Mar 2025 and accepted 22 Oct 2025.
* The author is with the Department of Mechatronics, College of Engineering, University of Mosul, Mosul 41002 Iraq.
E-mail: zmyousif@umosul.edu.iq, aws.anaz@uomosul.edu.iq , saeeds70@uomosul.edu.iq.
** The authors are with the Department of Computer, College of Engineering, University of Mosul, Mosul 41002 Iraq.
E-mails: basil.mahmood@uomosul.edu.iq
Corresponding Author: Zead Mohammed Yosif.

planning determines the target's collision-free route. Finally, motion control should be used to control the mobile robot's motion[3].

The most important thing is the existence of dynamic obstacles, which are obstacles in the environment that pose a danger to the robot. Actually, obstacles are at different positions, and they navigate in different directions and speeds. It is required for the robot to know the obstacles that exist within the environment and pose a threat to it. Another essential duty of the control system is to decide the speed and steering angle of navigation to avoid obstacles. The primary objective of this work is to design a module capable of enabling mobile robots to navigate around both dynamic and static obstacles. Additionally, the project emphasizes obstacle classification and incorporates the effect of relative velocity criteria in both obstacle classification and mobile robot control.

This work proposes a reactive navigation approach to deal with dynamic obstacle avoidance. The system consists of three main parts: the first part concerns finding the initial path by using the A* algorithm; the second part is for obstacle classification, which is achieved using a Neural Network; and the third part is represented by employing ANFIS for controlling the speed and steering angle to avoid dynamic obstacles. This system combines local and global navigation for mobile robots traveling in the indoor environment. This paper is organized as follows: section 1 presents the introduction. The relative works are included in section 2. In section 3, the proposed system is discussed and introduced. Section 4 presents the initial path algorithm. In section 5, the neural network and data collection. In section 6, mobile robot control using ANFIS is introduced. The results and discussion are introduced in section 7. Comparison with other work in section 8. The conclusion and suggestion are given in section 9.

2. Related work

As mentioned in the previous section, this work incorporates multiple intelligent techniques. Finding the initial path is a crucial step; hence, a widely recognized A* algorithm have been employed, which is known for its effectiveness in path planning [4]. Al- Arif et al. [6] Performed a comparison of three path planning algorithms, namely, the Breadth-first algorithm, the Dijkstra algorithm, and the A* (A-Star) algorithm, regarding computational time and memory usage. Among these, the A* algorithm yielded the most favorable results. Also, Giraldo [7] Conducted a comparison involving RPM, genetic algorithm, and A* algorithm, where the A* algorithm consistently produced the shortest path lengths.

Artificial Neural Networks (ANN) share many similarities with biological neural networks, and they are influenced by three factors: network design, weight generation (training/learning method), and the activation function. [8]. The Neural Network (NN) is used in many fields and applications, such as signal processing, image processing, pattern recognition, and mobile robot navigation. [9].

Jebur et al. [10] Proposed a Multi-layer Perceptron (MLP) for target recognition and obstacle avoidance. They utilized an IR camera to filter the red target from others and used MLP with IR sensors to avoid dynamic or static obstacles. A neural network was proposed by Aamer et al. [11]; they implemented an NN on an FPGA that uses pipelines. The NN was implemented on Xilinx Virtex-II to deal with the real-time experimental mobile robot with a dynamic obstacle.

In another study, Low et al. [12] Employed the Flower Pollination Algorithm (FPA) to improve Q-learning for finding paths in static environments. The combination of Q-learning and FPA, called IQ-FPA, resulted in an 11% increase in efficiency. Their experimental setup involved a three-wheel mobile robot operating in a 3x3 meter environment.

Khanisi et al. [13] A neural network produced the PWM of the right and left wheels from two input velocities. Li et al. [14] A neural network data fusion strategy was used to reduce the affection induced by inaccuracies in the environment or measurements and enhance the real-time performance and accuracy of localization for mobile robots in interior environments the position accurate within 6 cm. Pandey et al. [15] Introduced a path-planning optimization method using Particle Swarm Optimization (PSO) and Feed-Forward Neural Network (FNN). The FNN used distance sensors as inputs to calculate the steering angle and focused on path optimization. However, the use of PSO slowed down the system, the distance and time reduced about 8% and 9% respectively for time and distance. A summary of the papers mentioned above is provided in Table 1.

Table 1. NN research summary.

Paper	Method	Obstacle Type	Characteristics
[10] (2017)	MLP	Dynamic	Avoiding dynamic obstacle
[11] (2017)	NN	Dynamic	Use FPGA to implement NN, Pipelining, low memory, high-speed
[12] (2019)	IQ-FPA IQD	Static	IQ-FPA reduces computation time, and IQD produces a shorter and smoother path
[13] 2018	NN	Static	Controlling motors PWM
[14] 2020	NN	Static	Enhancing localization
[15] 2020	NN+PSO	Dynamic	PSO optimizes the path generated by NN

The integrated NN and Fuzzy Logic Inference system are the adaptive neuro-fuzzy inference system (ANFIS). Singh et al.[16] Suggested using ANFIS in mobile robots to avoid dynamic obstacles in unknown environments. The author employed three distance inputs (front, right, left) and an angle between them as ANFIS inputs, with the output being a suggested steering angle. The ANFIS system is called into action when the distance reaches a threshold that might lead to a collision between the robot and obstacles. By analyzing whether an object is close enough to cause a collision, the robot can effectively move away from obstacles in the opposite direction. The authors demonstrated the ANFIS's ability to control the mobile robot and avoid stationary and moving obstacles in crowded environments. Comparing to other paper the time and path reduced about 50%.

Pandey et al. [17] discussed the most crucial duties of any mobile robot: navigation and obstacle avoidance. The ANFIS controller was used for mobile robot navigation and obstacle avoidance in uncertain static environments. Several sensors, such as ultrasonic range finder and sharp infrared range sensors, are used to detect forward obstacles in the environment. Obstacle distances received from the sensors are fed into the ANFIS controller, and the controller output is a robot steering angle. NEAR and FAR are the two Gaussian linguistic variables chosen. They concluded that the ANFIS controller is superior in simulation and experimental testing. This proposed controller could solve dynamic obstacles and difficulties in the future.

For autonomous mobile robots' collision-free navigation, Gharajeh et al. [18] developed a model consisting of an ANFIS controller for local obstacle avoidance and a GPS-based controller for the robot's global navigation toward the target. The GPS-based controller maintains the robot's navigation direction toward the static or moving target. When the robot detects any obstructions nearby, the ANFIS controller determines a steering angle. It calculates the angle using the robot's left, front, and correct distances from obstacles.

Samadi et al. [19] proposed an effective path-planning strategy for autonomous collision-free navigation of wheeled mobile robots based on an Adaptive Neuro-Fuzzy Inference System (ANFIS). Three ultrasonic sensors installed on the robot's left, front, and right sides were utilized to estimate the distance between obstacles and the robot. The ANFIS-utility function block utilized these sensor distances as inputs to determine an obstacle avoidance steering angle for the robot—the suggested method produced paths roughly 30% shorter than those generated by other methods. Error! Not a valid bookmark self-reference.

Stavrinidis and Zacharia [20] introduce ANFIS to enhance a autonomous robot navigation system to achieve both static and dynamic obstacle. The authors compare between using ANFIS and fuzzy logic controller. The simulation results showed that the ANFIS lead to 33% rule reduction if compared to fuzzy logic controller in addition to reducing time to reach the destination about 2.5%.

Bede et al [21] propose a four navigation system , two of them for wheel speed controlling and the others for obstacle avoidance through suggestion steering angle.

Table 2 Summarizes the papers mentioned above.

Table 2. ANFIS papers summary.

Paper (year)	Method	Characteristics			
		Input	Output	Obstacle type	Simulation /experiment
[16] (2009)	ANFIS	Left and right front distances and target angle	Steering angle	Static + Dynamic	Experiment and simulation
[17] (2016)	ANFIS	Five distances	Steering angle	Static	Experiment and simulation
[18] (2020)	ANFIS	three distances	Steering angle	Static	Simulation
[19] (2022)	ANFIS	Three distances	Steering angle	Static	simulation
[20] (2024)	ANFIS	Six distances	Motor speed	Static + Dynamic	Simulation
[21] (2022)	ANFIS	Difference angle	Motor speed & steering angle	Static + dynamic	Simulation

In conclusion, this work must incorporate multiple intelligent techniques, including the widely recognized A* algorithm for path planning. Various studies on neural networks and fuzzy logic have demonstrated their effectiveness in mobile robot navigation, obstacle avoidance, and path optimization. Additionally, the proposed integrated approach, ANFIS, shows promising results in avoiding dynamic obstacles and achieving collision-free navigation. Training a system for obstacle danger classification and navigation controller produces faster and more efficient navigation in complex environments. The combination of the A* algorithm, neural networks, and ANFIS opens new possibilities for future mobile robot control and obstacle avoidance advancements.

3. Proposed system

The proposed system can be understood by examining the primary task of the mobile robot. The proposed system begins by initializing the environment and defining the mobile robot start and target position. The A* search algorithm is then employed to compute an initial, optimal path. While navigating, the system

continuously monitors dynamic obstacles to gather data on their speed and relative distance. This data is subsequently processed by BRNN which classified the obstacles. This classification is determined whether the obstacle is danger or not. I the obstacle is non-critical, the robot continuous along the original A* path. However, if an obstacle is classified as dangerous, an ANFIS is activated to adjust the robot speed and steering angle. After successful avoiding the obstacle, the robot returns to its originally planned path.

The proposed system is structured into three main components: (i) initial path generation using A* algorithm, (ii) obstacle collision prediction utilizing neural network, and (iii) obstacle avoidance implementation through ANFIS. the overall architecture of the proposed system is illustrated in Fig.1.

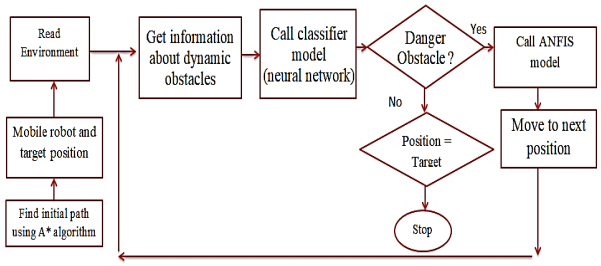


Fig.1: Proposed System Block Diagram.

4. Initial Path Planning

Problems with finding a path or path planning are widespread in robotics because it is critical for autonomous mobile robot navigation. The path-planning challenge for mobile robots remains an issue in autonomous robotic research. Robotic systems can choose an optimal or sub-optimal path based on one of the considered criteria of mobile robot motion (such as the lowest traveling cost, shortest route, shortest motion time, and so on) [22]. A suitable trajectory is created as a series of actions to keep the robot moving from the start state to the target point while passing through several intermediate states. Every decision made by path planning algorithms is based on the information available at the time and the criteria used [23]. The Dijkstra algorithm is the foundation for the A* algorithm [24]. The A* algorithm is one of the most widely used algorithms for determining a feasible path between two points. It is very effective and straightforward to put into practice. The A* algorithm solves path-planning problems in robotics and video games. When A* is used in a 2D path planning problem, the robot is treated as a point, and the radius of the mobile robot enlarges obstacles. This calculation can be done using configuration space calculation. A* algorithm obtains a path between the start and goal point by connecting four or eight connected nodes in a square shape, depending on the application. The A* algorithm is a best-first-search algorithm that combines the advantages of uniform-cost and greedy searches with a fitness function.

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

where g(n) denotes the total cost from the start node to the current node, and h(n) represents the heuristic estimated function from the current node to the goal node. Typically, h (n) is calculated using the Euclidean distance between the current node and the goal node. [25].

5. Obstacles classification and collision prediction

Classifying obstacles is very important, as it allows us to identify the degree of seriousness of the dynamic obstacles toward the mobile robot. The main objective of this classification is to determine whether the existing obstacle or obstacles are dangerous and have a probability of collision with the robot. two factors to assess obstacle danger have been suggested: the distance and the relative speed between the robot and the obstacle.

The most important question is how the robot can know the obstacles within the environment that pose a threat to it and may collide with it. One idea is that a closer obstacle is more dangerous than a far obstacle. Still, the main issue that should be considered is the speed of the obstacle because if there are two obstacles at the same distance from the robot, the more dangerous obstacle moves faster than the other. Depending these ideas, the areas surrounding the robot are categorized into five regions or zones. The five classified zones are Zone 1, Zone 2, Zone 3, Zone 4, and Zone 5. Depends on the relative velocity and distance between the mobile robot and obstacles, where the Zone 1 is considered the most dangerous, while Zone 5 is a safe area, and the degree of danger varies among the five regions. In this work, the relative velocity in addition to distance is added in the obstacle classifications. Table 3 summarizes the zone's characteristics.

Table 3. Zones Characteristics.

No	Zone Name	Speed (cm/sec)	Distance (cm)
1	Zone 5	<=4	>=90
2		<=6	>=120
3	Zone 4	8-10	>=120
4		6-8	90 to 120
5		1-4	60 to 90
6	Zone 3	>=10	>=90
7		8-10	90 to 120
8		6-8	60 to 90
9		<4	< 60
10	Zone 2	< 4	<60
11		8-10	60-90
12	Zone 1	>=10	<90
13		>=8	<60

5.1 Data collection

The region where obstacles surround the mobile robot can be divided into five zones. An essential step in collecting data is using it for neural network training so the neural network can classify the obstacles within the environment in which the robot moves.

For Zone 1, let the specified speed be 10 cm/sec and the distance between 30 and 60 cm. To cover the distances between 30 and 60 cm, the Zone is divided into 10 points or 10 readings (30, 33, 36, 39, 42, 45, 48, 51, 54, 57, 60). At each point, thirteen angle readings cover the range from -90 degrees to 90 degrees in a step of 15 degrees. The readings cover the range (-90) -(-75) -(-60) -(-45) -(-30) -(-15)-0-15-30-45-60-75-90. The idea is declared in Fig.2: Spectrum of path's angle, whereas the red region represents the area that needs to be divided. This case represents a speed of 10 cm/sec. The white arrows represent angle distribution. The dotted line between A&B points is the distance to be red. There are 10 readings at each angle, so each case has 130 readings. Also, there are 16 cases. Thus, there are 16 * 140 = 2080 samples. A Lidar sensor is used to measure the distance at certain angle.

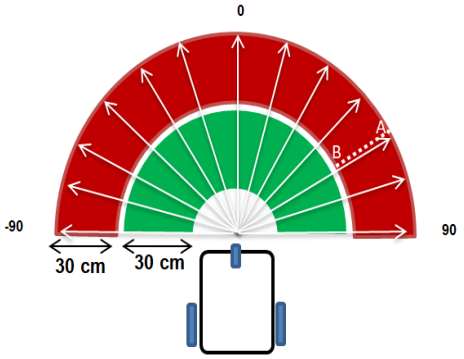


Fig.2: Spectrum of path's angle.

Integrating Matlab (2019) and CoppeliaSim (Edu V4.1.0) software, a robot simulation environment used to prototype, develop, and test robot systems and algorithms, achieves the data collection steps. [26]A distance sensor in CoppeliaSim has been used to read the distance of moving objects that move at a specific speed and angle. When the object moves toward the mobile robot, CoppeliaSim sends ten reading values to Matlab to collect data for the training process.

5.2 Neural network for obstacle classification

More than one method can be used to learn and train the data to create a specified model. A back-propagation (BP) neural network trains data and builds the module. The BP stands for error correction between output and target. The standard back-propagation neural network uses the BP algorithm, which has three types of layers: input layer, hidden layer, and output layer (Fig.3). The training data enters the neural network through the input layer; after that, the hidden layer represents the processing layer, and the output layer is the last stage that produces the module's decision [27]. The output layer's results are compared with target data to find the error between them. Suppose the error is large enough and more significant than the threshold value. In that case, the gradient descent method is used to update the weights' values along the connections in a backward manner from the output layer until they reach the input layer. This iterative process continues to reduce the error objective function Ew.

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

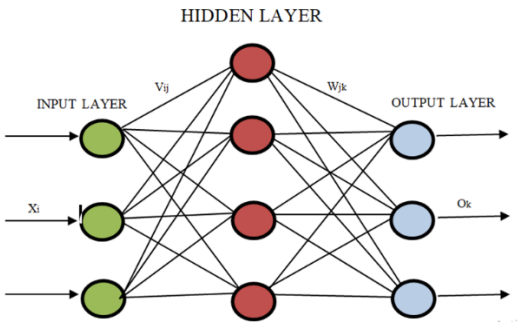


Fig.3: the multi-layer perceptron.

Where:

- x_i: The i^{th} input
- y_j: The output of the j^{th} hidden neuron
- O_k: The output of the k^{th} output neuron.
- t_k: The desired output
- V_{ij}: The weight from the i^{th} input to the j^{th} hidden neuron
- W_{kj}: The weight from the j^{th} hidden neuron to the k^{th} output neuron
- η: The learning rate.
- I: index for input neurons.

whereas

The hidden layer's weights are updated using the equation (3):

$$V_{ij}(n+1) = V_{ij}(n) + \eta \times \delta j(n) \times xi(n) \qquad (3)$$

δj is the error signal produced by the jth hidden neuron. The weights of the output layer are updated using the equation (4):

$$W_{kj}(n+1) = W_{kj}(n) + \eta \times \delta_k(n) \times y_j(n) \qquad (4)$$

δk is the error signal produced by the kth output neuron. As clear in equation (5)

$$\boldsymbol{\delta_k(n+1) = (t_k - O_k) \times (1 - O_k) \times O_k} \quad (5)$$

Using δk, the δj calculated from the equation (6) as follows:

$$\boldsymbol{\delta_j = (1 - y_j) \times y_j \times \sum_{k=0}^{k} \delta_k \times W_{jk}} \quad (6)$$

So, for any input, the output of the hidden neurons is obtained first, followed by the output of the output layer neurons. The above equations are used to update the weights of each hidden and output layer neuron. The hidden neuron error signals are propagated backward from the output layer to the hidden layer. This process is repeated for the next input-output pattern [28]. The process is repeated until the target error threshold or several iterations are reached, at which point learning stops. The descent of the standard BP learning algorithm, on the other hand, is a first-order gradient algorithm with a slow convergence rate.

The Matlab (trainbr) command is used to perform Bayesian regularization back-propagation. By learning the zones that were suggested based on two inputs, the best performance is used to generate the solution. There are approximately 2000 pieces of environmental data knowledge. The training data was divided using Matlab's default divide function. The data are randomly divided and applied to the neural networks. 70 % of the data is used for training to compute the gradient and update the network weight and biases. A validation set representing 15% of data is used to monitor errors during training. During the training process, the test set for plotting the test set error is 15% of the data. The network was trained over 1000 epochs.

The general structure of the neural network after data reduction is shown in Fig.4. From the confusion matrix of the Bayesian Regularization Neural Network, which has two Hidden Layers after data reduction, as shown in Fig.4, the training accuracy equals 99.1%. The trained neural network model is used to test samples of data. These sample outputs are known. The confusion matrix is shown in Fig.5. The characteristics of this network are summarized in Table 4.

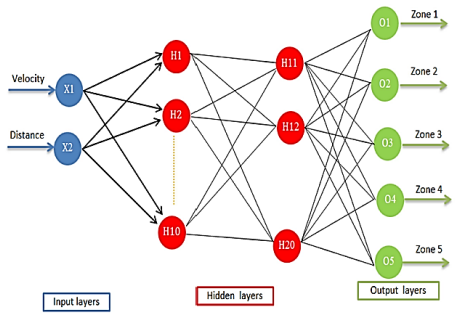


Fig.4: Two Hidden Layer Neural Network Structure After Data Reduction.

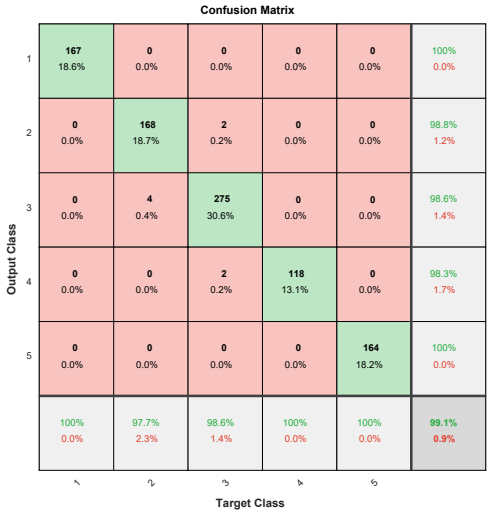


Fig.5: The Confusion Matrix of Bayesian Regularization Neural Network with Two Hidden Layers After Data Reduction.

Table 4: Bayesian Regularization NN with two hidden layers After Data Reduction.

	Characteristics	Neural network 2
1	Number of input layer neurons	2
2	Number of hidden layer neurons	10, 10
3	Number of output layer neurons	5
5	Hidden layer activation function	Bayesian Regularization
6	Output layer activation function	Linear
7	Learning rate	0,05
8	Maximum number of epochs	1000
9	Accuracy	99.1 %
10	Validation accuracy	99.1 %

6. Adaptive Neuro-Fuzzy Inference System for Collision Avoidance.

An ANFIS was suggested by Singh et al. [16] for mobile robot navigation to avoid dynamic obstacles. The ANFIS system combines fuzzy logic and a neural network, whereas the input layer is a fuzzy logic system. However, a neural network takes the fuzzy logic rule from trained data. By this method, the module takes advantage of fuzzy logic and neural networks by combining them and excluding their disadvantages.

The fuzzy logic approach is critical in the applications of intelligent systems, especially robotics. Integrating fuzzy systems and neural networks enhances the system performance by learning and updating membership function parameters to suit the environment. ANFIS combines a Neural Network and fuzzy logic system that uses empirical datasets to describe a complicated system's input/output behavior [18].

Takagi—Sugeno fuzzy inference system and artificial neural networks combine to form ANFIS. The ANFIS uses the dataset to build rules and a membership function. [17]. The ANFIS combines two fundamental machine learning techniques: the ANN and the Fuzzy Learning Machine (FLM). The FLM is utilized to transform the available inputs into desired outputs by utilizing densely linked ANN processing units. Early in the 1990s, it was developed as a universal estimator. In addition to combining the advantages of ANN and FIS, it also eliminates their disadvantages. It thus enhances generalization performance, archives extremely nonlinear mapping, and produces reliable solutions. As a result, researchers used it to handle classification, regression, and feature extraction problems in various fields of study. [29]. ANFIS's generalized structure comprises five layers (Fig.6) in addition to the input and output layers. [30].

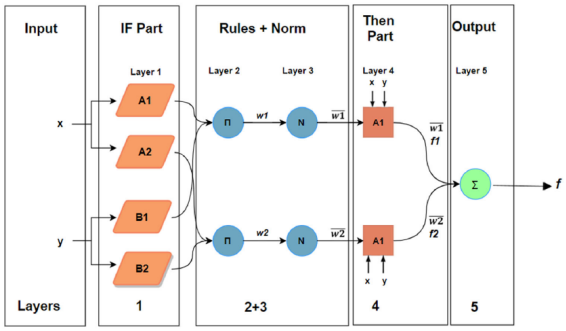


Fig.6: structure of ANFIS model with two inputs [31].

The characteristics of each layer are shown in the following:

Layer 1 (Fuzzyfing Layer): Neurons in this layer are adaptive nodes with premise parameters.

In layer 1, the outputs of the nodes A1 and A2 for the input x1 and the triangular membership function can be calculated at equation (7)

$$T_i^1(x_1) = \mu_{A_i}(x_1) = \max\left(\min\left(\frac{x_1 - a}{b - a}, \frac{c - x_1}{c - b}\right), 0\right); for\ i = 1,2 \quad (7)$$

Similarly, with the input x2, the outputs of nodes B1 and B2 can be expressed as clear at equation (8).

$$T_i^1(x_2) = \mu_{B_i}(x_2) = \max\left(\min\left(\frac{x_2 - a}{b - a}, \frac{c - x_2}{c - b}\right), 0\right); for\ i = 1,2 \quad (8)$$

Layer 2: (Implication Layer): The neurons are labeled Pi and are shown by a circle. The output node is then created based on the input signals. Equation (9) shows the output node indicates the firing strength of a rule wi.

$$T_i^2 = w_i = \mu_{A_i}(x_1) \times \mu_{B_i}(x_2); where\ i = 1,2 \quad (9)$$

Layer 3: (Normalizing Layer): Each neuron in this layer is a fixed neuron, denoted by a circle and designated N. The output is determined by the ratio of the ith rule's firing strength to the sum of all rules' firing strengths as it clear at equation (10) .

$$T_i^3 = \overline{w_i} = \frac{w_i}{\sum_{k=1}^{4} w_k}; where\ i = 1,2 \quad (10)$$

Layer 4: (Defuzzyfing Layer): The neurons in this layer are adaptive neurons with consequence parameters, see equation (11).

$$T_i^4 = \overline{w_i} t_i = \overline{w_i}(m_i x_1 + n_i x_2 + o_i); where\ i = 1,2 \quad (11)$$

Where mi, ni, and oi are the consequence parameters of the ANFIS model.

Layer 5: (Combining Layer): This layer comprises a single neuron that combines all the inputs as shown at equation (12).

$$T_i^5 = \sum_{i=1} \overline{w_i} t_i =; where\ i = 1,2 \quad (12)$$

The system has three inputs and two outputs. The inputs include relative robot-obstacle speed, distance, and angle between obstacle and robot, as shown in Fig.7.

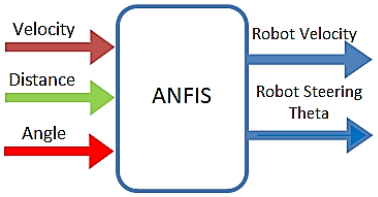


Fig.7: ANFIS Input/Output system.

The system outputs are suggested to be speed and the steering angle of the mobile robot. A data set should be available for system training to implement the ANFIS system. Five thousand records of data sets have been created. These data include a vector of 5 columns and 5000 rows; the first three variables of each row represent system inputs, while the last two variables include the outputs of speed and steering angle. The distribution training data cover a range of angles from -90- to 90 degrees, distances from 1 to 120 cm, and speeds from 1 to 50 cm/sec. Using ANFIS, up to n-inputs can acquired but cannot obtain more than one output. For this reason, two training systems are employed due to the presence of two outputs. The first training system is for speed, and the second system is for steering angle. The data set has been divided into three parts. The first part includes 4000 records for training, and the second and third parts

for testing and validation take about 500 samples each. The inputs of the two systems are the same. The speed training error is 0.71298, the testing error is equal to 1.0266, and the validation error is equal to 0.9582. The membership that is used here is Gaussian membership. The angle training error is 1.766, the testing error equals 3.93, and the validation error equals 3.08. The training, testing, and checking are presented in Fig.8 and Fig.9. The ANFIS block diagram is shown in Fig.10.

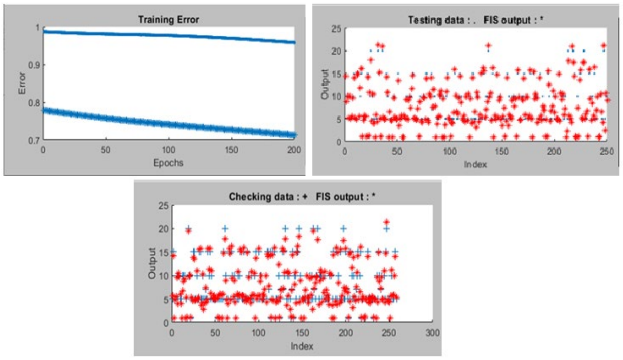


Fig.8: Adaptive Neuro-Fuzzy Systems Model for Mobile Robot Speed.

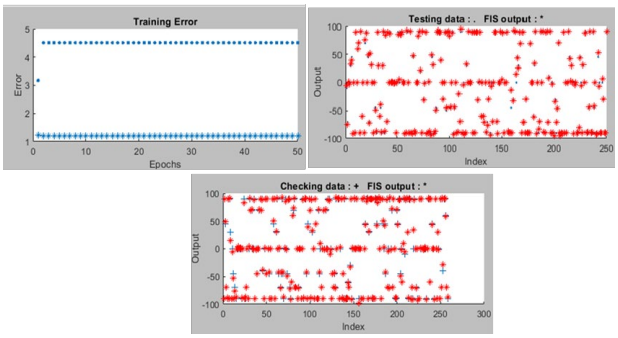


Fig.9: Adaptive Neuro-Fuzzy Systems Model for Mobile Robot Steering Angle.

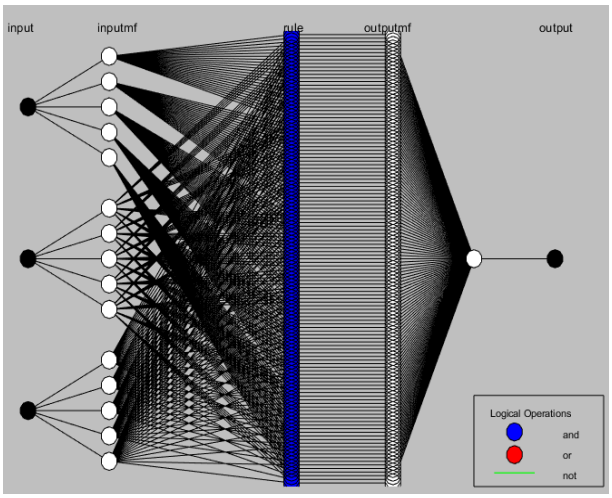


Fig.10: Adaptive Neuro-Fuzzy Systems Model Block Diagram.

7. Results and discussions

Environment chosen to implement the proposed system consists of 500 * 500 pixels. Each pixel represents 1 cm. Thus, the dimensions of the environment being worked on are 500x500 cm. These environment dimensions refer to the accuracy with which the robot moves, as the movement and accuracy will be at the level of one centimeter, which is good accuracy when moving the robot indoors[14] . Two types of environments have been used in this work. The first type is an empty environment that does not contain any static obstacles, which is called a free environment. The second proposed environment contains static obstacles. The positions of static obstacles have been chosen to make a challenge in some cases.

The distance was derived analytically from the spatial coordinates of two reference points – the robot and the obstacle. The distance measured using the Euclidian distance equation between two pints, also the relative angle between them.

The angle of the mobile robot can be calculated from two points, when the robot moves from one point to another, for example, P1 and P2 (the previous position and current position). We can use the tan inverse function to find the angle of the mobile robot, and in the same way, the angle of the obstacle can be found.

In the following scenario (Test 1), the mobile robot starts position from the left bottom environment with position (50,10). The target position is (400,400). As it is clear, this distance approximately covers the minor diagonal of the environment. There are many dynamic and static obstacles in the environment. Static obstacles were determined in the previous step by reading about the environment without dynamic obstacles. The static obstacle is represented by black bars in the environment. A dotted red line plots the initial path from the start position to the target. There are three dynamic obstacles in the environment. These dynamic obstacles move in different directions. The first obstacle is the red color. The obstacle speed is 10 cm/sec, moving in the right direction from the environmental center. The second obstacle is colored green color. The second dynamic obstacle moves at -50 degrees in the left-down section of the environment with a speed equal to 10 cm/sec. The third dynamic obstacle is blue. The position of this obstacle is at the top and moves down in a direction with a speed equal to 10 cm/sec. The current position of dynamic obstacles is referred to by colored circles (red, green, blue). A colored line indicates the distance traveled by the obstacle, and each line is colored according to the color of the obstacle. The robot is specified by using a black circle surrounded by a blue rectangle, which represents the robot's current location. As was mentioned, a red dotted line drew the proposed

path, while the path that was traveled by the robot is in black. Consider the suggested environment, including stationary and moving obstacles, and the recommended initial path. Any of the moving obstacles that pose a threat to the mobile robot can be classified through this data.

The suggested method continually gathers data on the robot's speed and the distance to the dynamic objects, so it continuously examines the moving obstacles and classifies them to find out which is more dangerous. As it is clear, the first dynamic obstacle is moving away from the mobile robot and its path. So, it is not classified as a danger or as having the possibility of collision. This moving obstacle is classified. After all, it is not dangerous because it's moving in a direction away from the robot. The second and third moving obstacles pose a danger to the moving robot, and there is a possibility of collision. For this reason, the system will decide to prevent a collision. The decision is represented by changing the speed of the mobile robot in addition to the steering angle.

The suggested angle and speed are the outputs of the adaptive neural fuzzy system, where the entries for this system are the angle between the robot and the obstacle, the relative speed between the obstacle and the robot, and the distance between the obstacle and the robot. These data are input into two models: the first gives the suggested speed of the mobile robot, and the second gives us the suggested angle at which the robot turns to avoid the dynamic obstacle. The flowchart of the proposed system is shown in Fig.11.

The robot avoids the obstacle twice; the first time, it avoids the second obstacle, and the second time, when the third obstacle is avoided. The distance of the proposed path is 540 cm, and the displacement between the starting point and the target is 530 cm, while the robot's length travelled after avoiding obstacles is 650 cm. The time taken by the robot from the start towards reaching the goal, including avoiding the two obstacles, is 8.8 seconds, and the number of iterations is 86. The information and characteristics that are obtained through this scenario are presented in Table 5 and Fig.12.

Another scenario (Test 2) is applied, including three dynamic obstacles. The first and the second dynamic obstacles are moving away from the mobile robot and its path, so they are not classified as dangerous obstacles or have the possibility of collision. Thus, these dynamic obstacles are classified. After all, they are not harmful because they move in a direction away from the robot; as for the third moving obstacle, even if it is moving in the direction of the proposed path of the robot, it is so far that it is classified on the basis that it is not dangerous. For these reasons, the robot continues motion on the initially suggested path.

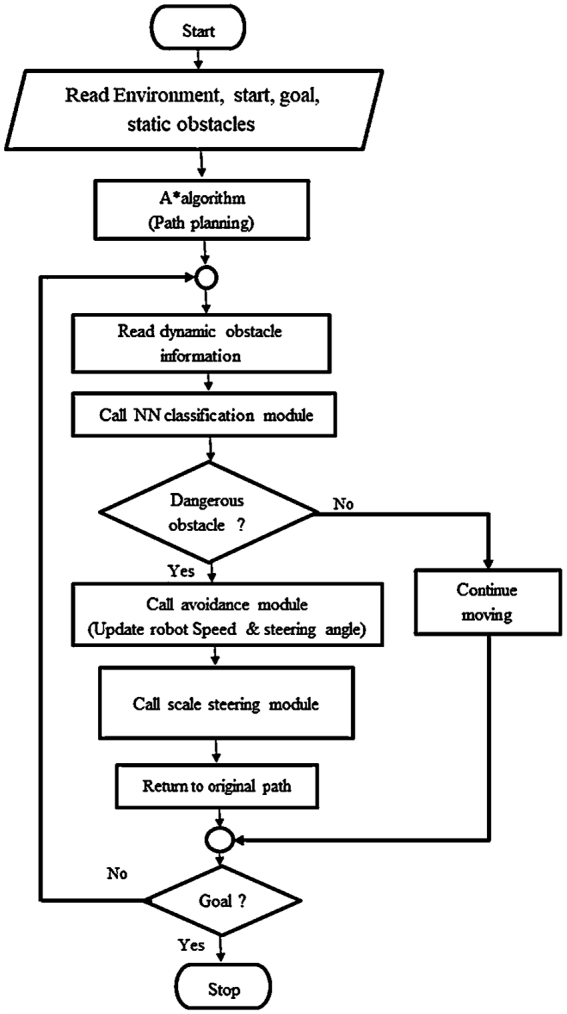


Fig.11: Proposed system flow chart.

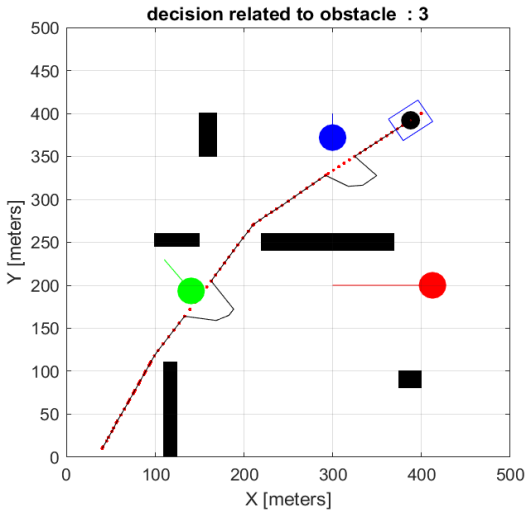


Fig.12: BRNN-ANFIS for Environment with Static and three Dynamic obstacles (Test 1).

The distance of the proposed path is 540 cm, and the displacement between the starting point and the target is 530 cm. The time taken by the robot from the beginning of the path until reaching the goal is 7.4 seconds. The number of iterations is 78. The information and characteristics are presented in Fig.13 and Table 5.

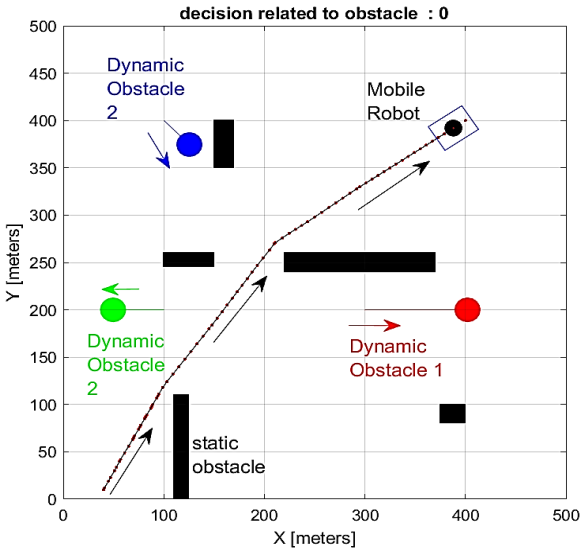


Fig.13: BRNN-ANFIS for environment with static and three safe dynamic obstacles (Test 2).

Another scenario (Test 3) applied here includes three dynamic obstacles. The first obstacle starts from the top right corner and moves toward the mobile robot. It starts from the left button corner with an angle of -140 degrees, and its speed equals 35 cm/sec. The second dynamic obstacle moves toward the mobile robot and its path with an angle of -45 degrees and 40 cm/sec speed. Obstacle 3 has the same speed as Obstacle 2, but the direction is -100 degrees.

When the three dynamic obstacles, they are seen to move towards the mobile robot, thereby posing a clear danger and the possibility of collision. Consequently, each of the three obstacles is classified as dangerous, and the obstacle avoidance is activated.

The critical aspect of this scenario is that the three dynamic obstacles move faster than in the previous examples. These obstacles pose a danger and the potential for a collision with the moving robot, but the robot avoids all these obstacles safely, well, and quickly.

The proposed path is 546 cm long, and the displacement between the starting point and the target is 530 cm. The robot traveled 660 cm after avoiding obstacles. The time taken by the robot from the beginning of the path to the goal is 16 seconds. The number of iterations is 116. Table 5 and Fig.14 show the information and characteristics obtained through the applied scenario.

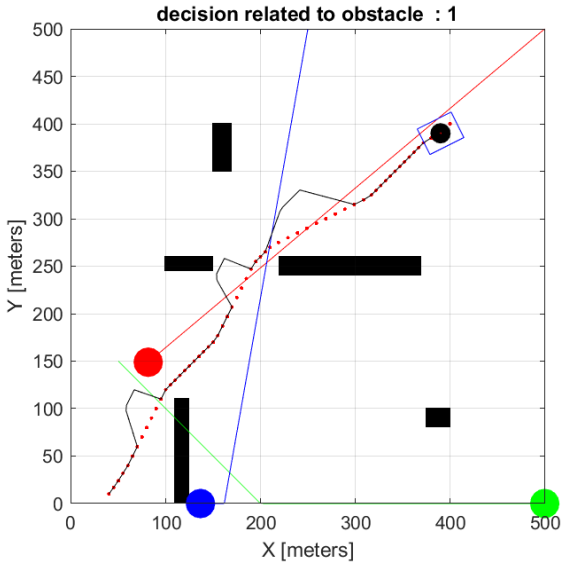


Fig.14: BRNN-ANFIS for Environment with Static and three dangerous Dynamic obstacles (Test 3).

A scenario (Test 4) applied here includes three dynamic obstacles. The first and second dynamic obstacles move toward the mobile robot and its path. Obstacle 1's speed is 10 cm/sec, and its direction is -150 degrees. Obstacle 2 and obstacle 3 have the same speed, which equals 10 cm/sec, and their directions are -37 and -90 degrees, respectively.

When the first obstacle is observed to be close to the robot, a decision is made for the obstacle to be avoided in order to prevent any collision. For this reason, the system classified obstacle one as dangerous. As shown in Fig.14, the mobile robot starts its avoidance at point A and returns to the path at point B. As soon as the mobile robot arrives at the initial path at point B, obstacle 2 is classified as dangerous or has the possibility of a collision. For this reason, the mobile robot moves away from obstacle 2. During the mobile robot traveling from point B, another obstacle (obstacle 3) moves closer to the mobile robot. The third obstacle direction is -90 degrees and moves toward the mobile robot. The most crucial issue in this scenario is continuously checking the environment during the path from point B to point C. In this situation, the mobile robot is using an avoidance strategy. Although the mobile robot uses the avoidance strategy at point C, the mobile robot system recalls the avoidance system again. After avoiding obstacle three, the mobile robot returns to point D's initial path toward the target. The distance of the proposed path is 300 cm, and the displacement between the starting point and the target is 300 cm. While the length that the robot travelled after avoiding obstacles was 350. The time taken by the robot from the beginning of the path until reaching the goal is 35 seconds. The number of iterations is 304. Table 5 and Fig.15 show the information and characteristics of the applied scenario.

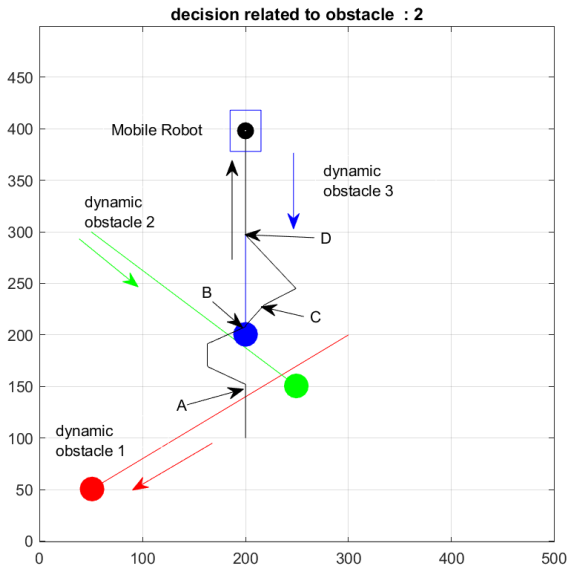


Fig.15: Applying BRNN-ANFIS to three dangerous dynamic obstacles (Test 4)

Table 5: Results and Characteristics for Tests (1-4).

Characteristics	Test 1	Test 2	Test 3	Test 4
Initial path length using A* (cm)	541	541	546	300
Time required for initial path(sec)	8	8	8	0.6
direct length from start to end points	530	530	530	300
Path length after avoidance (cm)	600	541	648	356
Obstacle 1 velocity(cm/s)	10	10	35	9
Obstacle 2 velocity(cm/s)	10	10	40	8
Obstacle 3 velocity(cm/s)	10	10	40	5
Robot velocity (cm /s)	120	120	60	10
Relative velocity 1 - (cm /s)	108	108	94	16.5
Relative velocity2 - (cm /s)	118	118	82	16
Relative velocity3 - (cm /s)	120	120	100	15
Time from start to goal	8 sec	7.4)	8 sec	35
No of iterations	78	78	115	78

The length of a paper is limited to 8 pages of two-

8. Comparison with other work

The proposed work was compared with two other existing works. The first comparison involved two scenarios compared with the proposed models. The first work under comparison suggested the use of a neural network and fuzzy logic (BRNN-FL) [27]. The comparison was based on several factors, including the distance the robot covered from the starting position to the target, the time to reach the target, and the number of iterations required to achieve the target.

The first comparison was conducted in the proposed environment (Test 1), which consisted of two dangerous moving obstacles and one safe obstacle amidst static obstacles. The results demonstrated an improvement in the system's performance. Notably, the time taken to reach the target was reduced by 27%, and the number of iterations needed decreased by 7%. Additionally, another test was carried out, this time involving three dangerous obstacles moving toward the robot. In this scenario, the proposed work achieved a time reduction of approximately 12%. The results of the first comparison are presented in Table 6.

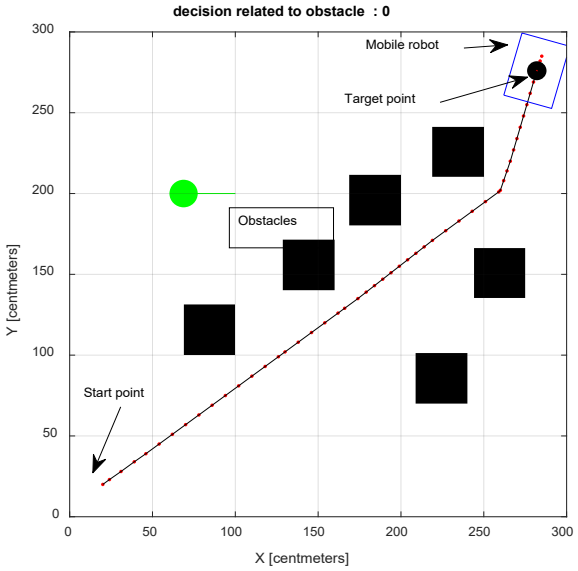
Table 6: Comparison of the proposed model with BRNN-FL [32].

Characteristics	BTNN-FL[32]	BRNN-ANFIS	BTNN-FL[32]	BRNN-ANFIS
Environment	Test 1		Test 4	
Initial path length using A*	541	541	300	300
Time required for initial path(sec)	8	8	0.6	0.6
direct length from start to end points	530	530	300	300
Obstacle 1 velocity(cm/s)	10	10	9	9
Obstacle 2 velocity(cm/s)	10	10	8	8
Obstacle 3 velocity(cm/s)	10	10	5	5
Robot velocity (cm /s)	120	120	10	10
Path length after avoidance (cm)	601	600	358	350
Increased path length ratio	11	11	19	17
Time from start to end (sec)	15	11	41	36
Number of iterations	92	86	321	304

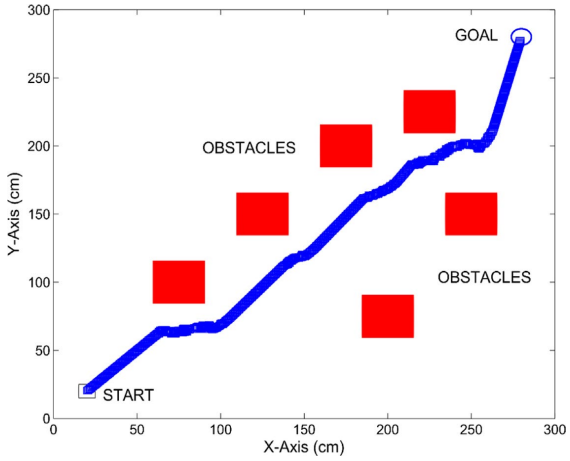
Another project is compared with the proposed model. Pandey et al. [15] Introduced a path planning optimization using PSO and FNN. They utilized a feed-forward neural network, with distance sensors as inputs and the steering angle as the output, to focus on path optimization. However, their implementation of PSO resulted in a system slowdown. Consequently, the robot's velocity was limited to 3 cm/sec, covering a distance of only 110 cm in 39 seconds. In contrast, our system allows for significantly higher speeds, reaching up to 100 cm/sec, which improves overall efficiency.

In comparison to our proposed model illustrated in Fig.16(a), the work by [15] was also examined, as depicted in Fig.16(b). The results of our designed model

exhibit superior performance in terms of both the distance travelled and the time taken to reach the goal, as evidenced in Table 7.



(A) Proposed method



(B) PSO-tuned FNN [15]

Fig.16: Comparison between BRNN-ANFIS (proposed method) and PSO-tuned FNN[15].

Our proposed system presents a significant advancement over the work in [20] by addressing its key limitations, namely the absence of global path planning, obstacle classification, and quantitative performance metrics. Our system introduces a hybrid intelligent architecture that integrate A* algorithm for optimal global path planning. The BRNN for classifying obstacle, and ANFIS for adaptive navigation control. Out system archives 6% reduction in path planning and 60 % faster navigation.

Table 7: Comparison and characteristics between BRNN-ANFIS and PSO-tuned FNN.

Characteristics	BRNN-ANFIS	PSO Tuned FNN[15]
Path length	387 cm	410 cm
Time required to travel	10 sec	39 sec
Time to find the initial path	5 sec	0
Total time	15 sec	39
Path length reduction	6%	
Time reduction	62%	

Based on previous comparisons, our proposed model outperforms existing approaches regarding navigation efficiency and obstacle avoidance. The comparisons show significant improvements in reaching the target faster and with fewer iterations. Our system's higher velocity capabilities offer a clear advantage over PSO and FNN-based methods. These findings highlight the effectiveness of our approach for mobile robot navigation in complex environments, making it a promising solution for future dynamic obstacle challenges.

9. Conclusion

Navigating an autonomous mobile robot with obstacles moving toward its path is still a significant challenge. The use of a single-stage module leads to a deficiency and a limitation in the controller's performance in avoiding dynamic obstacles.

The proposed dynamic obstacle's data collection of relative speeds and distances for training neural networks made it capable of making correct zone classification. Using the neural network to classify the danger of the moving obstacle leads to reducing the processing time. Thus, the mobile robot could rush in the presence of safe, dynamic obstacles and fast dynamic obstacles.

The simulation results proved that using the ANFIS to control the mobile robot's speed and direction of motion allows it to deal with multiple obstacles at different speeds; this achievement is attributable to the controller's response speed based on the ANFIS.

The proposed work is compared to state-of-the-art research papers. In the first comparison, the use of BRNN-ANFIS showed better performance in terms of time, which was reduced by 27% and 12% for the two tested scenarios, respectively. BRNN-ABFIS was compared with another work that used NN only in the second comparison. The proposed work demonstrated better performance in path length reduction (approximately 6%) and time taken reduction to reach the target, which is reduced by about 60%.

The module used for fixed or stationary target, for future a dynamic target should be considered, also multi target have to be added to the environment.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

The authors have accepted the responsibility for the entire content of this manuscript and approved its submission.

Funding

No funding was received for this work.

Informed Consent Statement

Not applicable.

Declaration of generative AI and AI-assisted technologies

The authors declare that no generative AI or AI-assisted technologies were used in the writing process of this manuscript.

Acknowledgment

The authors would like to thank the faculty and employees of the University of Mosul / College of Engineering, particularly the Mechatronics engineering and Computer engineering departments, for their assistance in completing this study.

References

[1] P. Martin and A. P. Del Pobil, 'Application of artificial neural networks to the robot path planning problem', WIT Transactions on Information and Communication Technologies, vol. 6, 2025.

[2] E. A. Rodríguez-Martínez, W. Flores-Fuentes, F. Achakir, O. Sergiyenko, and F. N. Murrieta-Rico, 'Vision-Based Navigation and Perception for Autonomous Robots: Sensors, SLAM, Control Strategies, and Cross-Domain Applications—A Review', Eng, vol. 6, no. 7, p. 153, Jul. 2025, doi: 10.3390/eng6070153.

[3] Z. Yosif, B. Mahmood, and S. Al-khayyt, 'Assessment and Review of the Reactive Mobile Robot Navigation', (AREJ), vol. 26, no. 2, pp. 340–355, Oct. 2021, doi: 10.33899/rengj.2021.129484.1082.

[4] Y. Zhang, C. Cui, and Q. Zhao, 'Path Planning of Mobile Robot Based on A Star Algorithm Combining DQN and DWA in Complex Environment', Applied Sciences, vol. 15, no. 8, p. 4367, Apr. 2025, doi: 10.3390/app15084367.

[5] F. Duchoň et al., 'Path Planning with Modified a Star Algorithm for a Mobile Robot', Procedia Engineering, vol. 96, pp. 59–69, 2014, doi: 10.1016/j.proeng.2014.12.098.

[6] S. Al-Arif, A. Ferdous, and S. H. Nijami, 'Comparative study of different path planning algorithms: a water based rescue system', International Journal of Computer Applications, vol. 39, 2012.

[7] B. Hernández and E. Giraldo, 'A Review of Path Planning and Control for Autonomous Robots', in 2018 IEEE 2nd Colombian Conference on Robotics and Automation (CCRA), Nov. 2018, pp. 1–6. doi: 10.1109/CCRA.2018.8588152.

[8] X. Xu and N. Gupta, 'Application of radial basis neural network to transform viscoelastic to elastic properties for materials with multiple thermal transitions', J Mater Sci, vol. 54, no. 11, pp. 8401–8413, Jun. 2019, doi: 10.1007/s10853-019-03481-0.

[9] S. Shanmuganathan, 'Artificial neural network modelling: An introduction', in Artificial neural network modelling, Springer, 2016, pp. 1–14.

[10] M. H. Jebur and M. M. Ali, 'Safe navigation and target recognition for a mobile robot using neural networks', in 2017 14th International Multi-Conference on Systems, Signals & Devices (SSD), IEEE, 2017, pp. 705–712.

[11] N. Aamer and S. Ramachandran, 'Neural network, VLSI approach for autonomous robot navigation', in 2017 2nd International Conference on Communication and Electronics Systems (ICCES), Coimbatore: IEEE, Oct. 2017, pp. 935–942. doi: 10.1109/CESYS.2017.8321220.

[12] E. S. Low, P. Ong, and K. C. Cheah, 'Solving the optimal path planning of a mobile robot using improved Q-learning', Robotics and Autonomous Systems, vol. 115, pp. 143–161, May 2019, doi: 10.1016/j.robot.2019.02.013.

[13] K. Khnissi, C. Seddik, and H. Seddik, 'Smart Navigation of Mobile Robot Using Neural Network Controller', in 2018 International Conference on Smart Communications in Network Technologies (SaCoNeT), IEEE, 2018, pp. 205–210.

[14] H. Li, Y. Mao, W. You, B. Ye, and X. Zhou, 'A neural network approach to indoor mobile robot localization', in 2020 19th International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES), Oct. 2020, pp. 66–69. doi: 10.1109/DCABES50732.2020.00026.

[15] A. Pandey, V. S. Panwar, M. E. Hasan, and D. R. Parhi, 'V-REP-based navigation of automated wheeled robot between obstacles using PSO-tuned feedforward neural network', Journal of Computational Design and Engineering, vol. 7, no. 4, pp. 427–434, Aug. 2020, doi: 10.1093/jcde/qwaa035.

[16] M. K. Singh, D. R. Parhi, and J. K. Pothal, 'ANFIS Approach for Navigation of Mobile Robots', in 2009 International Conference on Advances in Recent Technologies in Communication and Computing, Kottayam, Kerala, India: IEEE, 2009, pp. 727–731. doi: 10.1109/ARTCom.2009.119.

[17] A. Pandey, S. Kumar, K. K. Pandey, and D. R. Parhi, 'Mobile robot navigation in unknown static environments using ANFIS controller', Perspectives in Science, vol. 8, pp. 421–423, Sep. 2016, doi: 10.1016/j.pisc.2016.04.094.

[18] M. S. Gharajeh and H. B. Jond, 'Hybrid Global Positioning System-Adaptive Neuro-Fuzzy Inference System based autonomous mobile robot navigation', Robotics and Autonomous Systems, vol. 134, p. 103669, Dec. 2020, doi: 10.1016/j.robot.2020.103669.

[19] M. Samadi Gharajeh and H. B. Jond, 'An intelligent approach for autonomous mobile robots path planning based on adaptive neuro-fuzzy inference system', Ain Shams Engineering Journal, vol. 13, no. 1, p. 101491, Jan. 2022, doi: 10.1016/j.asej.2021.05.005.

[20] S. Stavrinidis and P. Zacharia, 'An ANFIS-Based Strategy for Autonomous Robot Collision-Free Navigation in Dynamic Environments', Robotics, vol. 13, no. 8, p. 124, Aug. 2024, doi: 10.3390/robotics13080124.

[21] D. Patel and K. Cohen, 'Obstacle Avoidance and Target Tracking by Two Wheeled Differential Drive Mobile Robot Using ANFIS in Static and Dynamic Environment', in Fuzzy Information Processing 2020, vol. 1337, B. Bede, M. Ceberio, M. De Cock, and V. Kreinovich, Eds., in Advances in Intelligent Systems and Computing, vol. 1337. , Cham: Springer International Publishing, 2022, pp. 337–348. doi: 10.1007/978-3-030-81561-5_28.

[22] H. Liu, 'Chapter 1 - Introduction', in Robot Systems for Rail Transit Applications, H. Liu, Ed., Elsevier, 2020, pp. 1–36. doi: 10.1016/B978-0-12-822968-2.00001-2.

[23] A. Montazeri, A. Can, and I. H. Imran, 'Chapter 3 - Unmanned aerial systems: autonomy, cognition, and control', in Unmanned Aerial Systems, A. Koubaa and A. T. Azar, Eds., in Advances in Nonlinear Dynamics and Chaos (ANDC). , Academic Press, 2021, pp. 47–80. doi: 10.1016/B978-0-12-820276-0.00010-8.

[24] A. M. Elshaer, R. A. Elmanfaloty, E. Abou-Bakr, M. Elrakaiby, and K. Saada, 'Exploring Algorithmic Efficiency of A-Star and Dijkstra for Optimal Route Planning in Green Transportation', Int. J. ITS Res., vol. 23, no. 2, pp. 1097–1107, Aug. 2025, doi: 10.1007/s13177-025-00503-x.

[25] N. Sariff and N. Buniyamin, 'An Overview of Autonomous Mobile Robot Path Planning Algorithms', in 2006 4th Student Conference on Research and Development, Jun. 2006, pp. 183–188. doi: 10.1109/SCORED.2006.4339335.

[26] B. Bogaerts, S. Sels, S. Vanlanduit, and R. Penne, 'Connecting the CoppeliaSim robotics simulator to virtual reality', SoftwareX, vol. 11, p. 100426, Jan. 2020, doi: 10.1016/j.softx.2020.100426.

[27] D. Wu et al., 'Application of Bayesian regularization back propagation neural network in sensorless measurement of pump operational state', Energy Reports, vol. 8, pp. 3041–3050, Nov. 2022, doi: 10.1016/j.egyr.2022.02.072.

[28] N. H. Singh and K. Thongam, 'Neural network-based approaches for mobile robot navigation in static and moving obstacles environments', Intel Serv Robotics, vol. 12, no. 1, pp. 55–67, Jan. 2019, doi: 10.1007/s11370-018-0260-2.

[29] M. Shafiullah, M. A. Abido, and A. H. Al-Mohammed, 'Artificial intelligence techniques', in Power System Fault Diagnosis, Elsevier, 2022, pp. 69–100. doi: 10.1016/B978-0-323-88429-7.00007-2.

[30] Y.-H. Chen and C.-D. Chang, 'An intelligent ANFIS controller design for a mobile robot', in 2018 IEEE International Conference on Applied System Invention (ICASI), Chiba: IEEE, Apr. 2018, pp. 445–448. doi: 10.1109/ICASI.2018.8394280.

[31] D. J. Armaghani and P. G. Asteris, 'A comparative study of ANN and ANFIS models for the prediction of cement-based mortar materials compressive strength', Neural Comput & Applic, vol. 33, no. 9, pp. 4501–4532, May 2021, doi: 10.1007/s00521-020-05244-4.

[32] Z. M. Yosif, B. S. Mahmood, and S. Z. Saeed, 'Artificial Techniques Based on Neural Network and Fuzzy Logic Combination Approach for Avoiding Dynamic Obstacles', JESA, vol. 55, no. 3, pp. 339–348, Jun. 2022, doi: 10.18280/jesa.550306.

Biographies

Zead Mohammed Yosif received the B.Sc. degree in Computer Engineering from University of Mosul, Mosul, Iraq, in 2006, and the M.Sc. degree from Cankaya University, Turkey. He got the Ph.D. degree in Computer engineering Department, College of Engineering, University of Mosul in 2022. He is a lecturer at Mechatronics Engineering Department, University of Mosul.

Basil Sh. Mahmood was born in 1953 in Mosul/ Iraq, graduated in 1976 from the University of Mosul/ Electrical department, and the M.Sc. degree in Electronics and Communications in 1979. Then he joined in Computer Center of the same university as an assistant lecturer then after he got the degree of Ph.D. On microprocessors architecture in 1996. Now, he is a microprocessors and computer architecture professor in the Computer Engineering Department/the University of Mosul. He published with others four books and more than 50 research papers in many journals and conferences.
He supervised more than 22 M.Sc. and 14 Ph.D.
Students. His interests are in microprocessors, computer architectures, image and signal processing, modern methods of Artificial Intelligence. He awarded many prizes and Medals.

Saad Zaghlul Al-khayyt received the B.Sc. and M.Sc. degrees in Mechanical engineering from Al-Nahreen University, in 1992, and 1995 respectively, Baghdad, Iraq, and the Ph.D. degree in Mechanical engineering from Russia. He has been a professor of Mechatronics engineering in 2025. He has many published papers in the field of mechatronics and robotics.

Aws Anaz received his Ph.D. in Doctor of Philosophy in Electrical and Computer Engineering from the University of Missouri-Columbia in 2019, where he specialized in smart medical devices. He is currently an instructor in the Mechatronics Department at the University of Mosul. Besides his academic experience, he has spent some years working in the industry as a field engineer. His current research interests include intelligent systems design, medical devices, and rehabilitation robots. Recent work has investigated automated screening of patients recovering from hand surgery