Robotica: (2020) volume 38, pp. 1627–1641. © Cambridge University Press 2019
doi:10.1017/S0263574719001668

Trajectory Planning and the Target Search by the Mobile Robot in an Environment Using a Behavior-Based Neural Network Approach

Krishna Kant Pandey* and Dayal R. Parhi

Mechanical Engineering Department, National Institute of Technology, Rourkela, India

(Accepted October 19, 2019. First published online: November 14, 2019)

SUMMARY

Navigation and path analysis in a cluttered environment is a challenging task over the last few decades. In this paper, a behavior-based neural network (BNN) and reactive control architecture have been presented for navigation of the mobile robot. Two different reactive behaviors have been taken as inputs function. Obstacle position is the first reactive behavior given by $u(o)$, whereas obstacle angle $u(n)$ according to the target position is the second reactive behavior. The angular velocity and steering angle are the output of the controller. The backpropagation architecture reduces the errors of weight function and records the best weight data that match the BNN controller. Using the BNN algorithm, the robot reacts quickly as compared to other developed techniques. To validate the performance of the controller, simulation and experimental results have been compared in the common platforms. The deviation in results for both the scenarios is found to be within 10%. The results of the BNN algorithm have also been compared with other existing techniques. Effectiveness of the proposed technique is measured in terms of smoothness of the realistic path, collision point detection, path length, and performance time.

KEYWORDS: Behavior-based neural network (BNN); Navigation; Path planning; Obstacle avoidance; Robot localization.

1. Introduction

To develop a robust navigational control algorithm for the mobile robot, a vast number of researches have been carried out in the last decade. The majority of the research defines a particular problem state. The aim of the mobile robot navigation system is to design and mainframe a mobile-based platform that helps to execute the tasks from start to the goal autonomously and robustly. The output of navigational control enhances the ability of the mobile robot, to plan and explain practically collision-free motion within the environment, where the environment may be rough, nonstructured, and maze. To enhance the ability of the navigational control system, a robust intelligent technique is required. The implementation of artificial intelligent (AI) technique helps the robot to understand the structures of the environment and execute the navigational tasks accordingly. Literature on different events, which have been performed by the mobile robot during the target search, has been presented in the upcoming paragraph.

Fierro and Lewis[1] presented a research paper on control of a nonholonomic[2] mobile robot using a neural network (NN). They have shown a combination of kinematic and torque control law developed using backstepping and stability theory that includes Lyapunov concept. An idea on multisensor system integration for positioning and navigation instruments using an artificial NN is proposed

* Corresponding author. E-mail: kknitrkl@yahoo.in

by ref. [3]. In this analysis, the inertial navigation system and GPS have been integrated over the Kalman filter to find the best solution during positioning and velocity calculation of real-time vehicle. Araujo et al.[4] presented the full integration of compact educational mobile robotic platform built using an Arduino controller board with robot operating system (ROS). The ROS provided by driver enables the use of several tools for data analysis in a mobile platform. Marin et al.[5] emphasized local fusion technique to improve the localization of a mobile robot. It provides the faster execution of data for the robot when compared to more complex fusion schemes. Wei et al.[6] presented a vehicle positioning technique using stereoscopic system, laser range finder, and GPS for localization of mobile agents inside the search space. Unscented information filter (UIF) combined with sensors to localize the vehicle and tested with real-time data to validate the real-time kinematic (RTK)-GPS data in ground level. Mujahed et al.[7] proposed the concept of admissible gap (AG) methodology for reaction-based collision avoidance.[8] This AG methodology gave direct shapes and kinematics to the robot to target the goal. Gualda et al.[9] proposed a simultaneous calibration and navigation algorithm using multiple ultrasonic local positioning systems. This proposal is used to navigate the mobile robot by estimating the map. The algorithm is based on multiple filters running in a parallel manner that estimates the global and local trajectories of a mobile robot. Khan et al.[10] designed an algorithm for autonomous navigation of a mobile robot in an indoor environment. They have used ultrasonic sensor and rational odometer to avoid obstacles and localize the robot, respectively. To improve the navigation and obstacle avoidance, the authors have used Bug-2 algorithm. Mota et al.[11] discussed the use of radio-frequency identification (RFID) technology to ensure the position of the robot inside the search space. Petri Net[12] dynamics model has been used as the navigation and perception tools. They used cards with RFID technology, which are placed at each intersection of the structured environment in ways that transform the information to each other during operation. Toth et al.[13] developed a fuzzy-based indoor navigation for mobile robots using Mamdani architecture. They did the relative study between developed method and other techniques without conducting real-time experiment. Xu et al.[14] proposed a multisensory self-supervised framework for navigation of mobile robot in the indoor environment.[15] To improve human supervision during operation, they combined the sensor policy with recording policy to evaluate accuracy in tasks during navigation without human interface. Navigation using sensor fusion data (acceleration, angular rate, and geometric fields attitude) collected by the mobile robot was proposed by Lee et al.[16] in the outdoor environment. In this analysis, magnetically disturbed environment (ferromagnetic objects) has also been considered during navigation of the mobile robot using compensated heading angle methodology. The Kalman filter algorithm[17] was developed in a way that detects and rejects the magnetic disturbances of the geometric field. In the case of heading accuracy, the peak-to-peak errors were reduced by 32.9%. To collect the garbage on the grass, Bai et al.[18] designed a novel garbage pickup robot. Deep learning NN control architecture was used to develop the controller. Using ground segmentation-based NN controller, the robot detects the garbage autonomously on the grass. Experimental results presented the effectiveness (garbage recognition rate 95%) of the controller without path planning. Speck et al.[19] explored the application of Shakey the robot that had an impressive capability of navigation without mobile manipulation. In this research, authors rebuilt Shakey with modern robotics technology. The rebuilt Shakey 2016 system has been applied on real robotics platform by testing with object rearranging trials. Kunze et al.[20] presented a survey based on AI for long-term robot autonomy. Autonomous agent, AI-based method, and long-term autonomy were the main objectives of this survey. They further categorized the application of robotics, which was used previously in the field of space, marine, air, and road services. Lui and Sukhatme[21] modified the standard Markov decision process to time-varying Markov decision process (TVMDP) and proposed its capability to handle future transition model over the horizon. To validate the framework, a marine robotics platform was used with spatiotemporal ocean data. The Bellman backup mechanism and Kolmogorov equation were used for computing work with TVMDP. Pierson et al.[22] proposed path planning for multiple agents in a bounded convex environment. They used a distributed AI algorithm and area minimization technique for the cooperative search of multiple evades and multiple followers. 2D and 3D environment were taken to conduct the simulation experiments. Based on the comparison results, it was concluded that the human controlled evader is unable to avoid capture, whereas autonomous evader negotiated with capture. Zhang et al.[23] developed the automated base path planning architecture for the industrial robot[24] using improved Rapidly Exploring Random Tree algorithm in a complex environment. Over-search-creation and unnecessary

https://doi.org/10.1017/S0263574719001668 Published online by Cambridge University Press

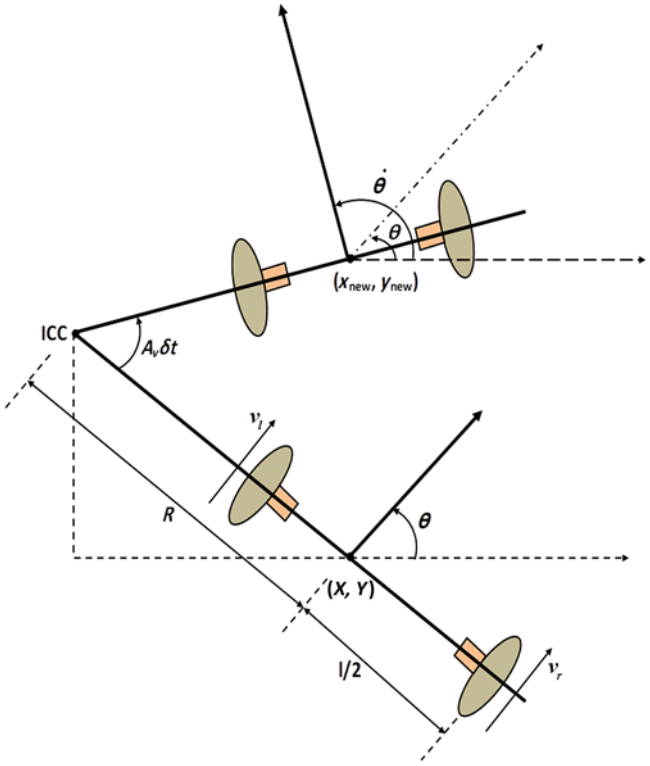

Fig. 1. The robot changes pose from the initial position to the new position.

iteration were eliminated, and search was refined using regression mechanism. Arango et al.[25] developed and implemented the hyperspherical algorithm for path planning of terrestrial robot. The path planning events were conducted using the homotopy continuation method (HCM). Parameters that affect the behavior of HCM was discussed and opted optimum values. Based on the experiments, the proposed methodology turned out to be faster when compared to other Spontaneous Bacterial Peritonitis algorithm. Motivation part of literature is not limited to safe navigation and shortest path planning using AI techniques having different sensory modules. However, how to adapt and learn from the environmental events is another required study field for mobile robotics research, which is already discussed in the literature section.

2. Dynamic Modeling of the Wheeled Mobile Robot

In Fig. 1, the motion of mobile robot has been shown in the global framework. Velocity (v) defines the velocity of the robot toward the target. The robot changes local position from pose (x, y) to the global pose (x_{new},y_{new}) using local theta (θ) angle to the global theta dot ($\dot{\theta}$) angle. At that time, angular velocity (A_v) changes with from 't' to 't + δt' and the velocity of wheels changes independently. Now, the velocity of left and right wheels can be expressed as:

$$v_l = A_v\left(R - \frac{l}{2}\right) \text{ or } v_r = A_v\left(R + \frac{l}{2}\right) \tag{1}$$

where 'A_v' is the angular velocity of wheels and 'R' is the radius from the Instantaneous Center (ICC) to the local center point of the robot given in Fig. 1. To calculate the values of 'R' and 'A_v,' Eqs. (2) and (3) have been used.

$$R = \frac{l}{2}(v_l + v_r) \div (v_r - v_l) \tag{2}$$

$$A_v = (v_r - v_l) \div l \tag{3}$$

If the robot rotates wheels according to the 'ICC' during the motion then robot local reference time 't' changes with time 'δt'. Now, the position of the robot is updated and given by new local pose on

https://doi.org/10.1017/S0263574719001668 Published online by Cambridge University Press

the global pose. Similarly, theta (θ) can be updated by theta dot ($\dot{\theta}$) and its value is calculated using Eq. (4):

$$\dot{\theta} = A_v \delta t + \theta \tag{4}$$

The center of rotation for the mobile robot according to 'ICC' is given by Eq. (5):

$$ICC = [ICC_X, ICC_Y] = \left[x - \frac{l}{2} (v_l + v_r) \div (v_r - v_l) \sin\theta, Y + \frac{l}{2} (v_l + v_r) \div (v_r - v_l) \cos\theta \right] \tag{5}$$

Now, the updated pose of the mobile robot from time 't' to '$t + \delta t$' in 'X' and 'Y' direction with rotation is calculated using Eqs. (6) and (7).

$$x_{new} = \cos(A_v \delta t)(x - X_{ICC}) - \sin(A_v \delta t)(y - Y_{ICC}) + X_{ICC} \tag{6}$$

$$y_{new} = \sin(A_v \delta t)(x - X_{ICC}) + \cos(A_v \delta t)(y - Y_{ICC}) + Y_{ICC} \tag{7}$$

where

$$ICC_X = x - R_{new} \sin\theta \quad or \quad ICC_y = y + R_{new} \cos\theta \tag{8}$$

$$A_v \delta t = (n_r - n_l)\ steps \div l \quad or \quad R_{new} = \frac{l}{2} (n_l + n_r) \div (n_r - n_l) \tag{9}$$

$$\varphi = \tan^{-1} \left[\frac{Tar_y - R_{y_{new:n\_steps}}}{Tar_x - R_{x_{new:n\_steps}}} \right] \tag{10}$$

In Eq. (9), 'n' is the counter value of the encoder and n: n steps equal to $v\delta t$, now 'v' becomes $n\_steps \div \delta t$. Using Eq. (10), the robot decides the angle of steering toward the target. In the next section, behavior-based navigation approach has been presented using NN.

3. Robot Navigation Using Behavior-Based NN

In this analysis, unknown and wall type obstacle arena has been taken under consideration for the study of path planning and navigation of a mobile robot. A behavior-based neural network (BNN) methodology has been used for robot navigation. In a series, the robot has used two behaviors as inputs. These behaviors have been taken in terms of obstacle distance from the robot position and obstacle angle from the robot position according to the goal position. Based upon BNN analysis, the robot has used the minimum threshold value to avoid obstacles and holds the goal position using negotiating obstacle angle. To minimize the path length according to the goal position and computational time within the environment is the objective of this analysis. BNN has been implemented as a control algorithm for the robot localization and navigation. BNN is a computational structure inspired by the behavioral study of biological neural processing. Using the BNN technique, the mobile robot learns from different environmental activities and executes the collision-free path in the working space from the start to the target point smoothly. The designed navigation control algorithm is based on the trained network. Figure 4 shows the training procedure of the mobile robot and technique that has been used for the proposed analysis. In the next section, the architecture of BNN technique has been presented. In Fig. 3, $u(0)$ and $u(n)$ are the two inputs taken using sensory module and explored by BNN.

3.1. The architecture of BNN

The NN is a combination of weights and strings as the human brain connected with the neurons and their connecting links. Generally, the NN comprises of three-layer combination of weights, input layer, hidden layer, and the last layer is called the output layer, from where we find the output data from the NN. A complete working principle methodically of NN has been shown in Fig. 2.

The processing essences are seen as entities that are akin to the neurons in a human brain and therefore, they are referred as cells, neuromas, or artificial neurons. The weights used on networks between dissimilar layers have significance in the working of the NN and the categorization of a network. There are two actions possible in the NN. First is, start with one set of weights and run the network without training. Second possible action is, start with one set of weights, run the network,

https://doi.org/10.1017/S0263574719001668 Published online by Cambridge University Press

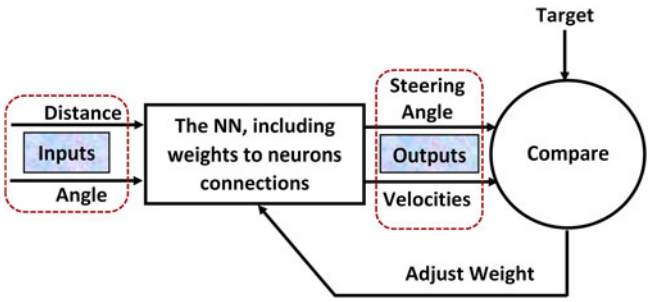

Fig. 2. Simple flow diagram of a neural network.

and modify some or all weights. Again, run the network with a new set of weights for accuracy. Repeat this process until the target is reached. In NN, updation of weight is called a training process of the network. Changing of weight inside the system takes place internally and rapidly, due to the desired learning of network. The most important thing in the design and development of the intelligent mobile system is the navigation problem.

3.2. Mathematical modeling of NN

Status of the neurons is determined by using Eq. (11),

$$S_k = \sum_{i=1}^{k} u_i \bullet Wt_i + G \tag{11}$$

where Wt_i is the weight of a neurons, $i = 1, 2, 3 \ldots\ldots k$; G is an offset value; S_k is the result of summation; and u_i is the element of the input vector (input), $i = 1, 2, 3 \ldots\ldots k$ of the neurons. Learning ability during operation is the key of the BNN and behind its use. To train the mobile robot, for smooth navigation inside the environment, an intelligent robot control algorithm is required. Therefore, the modified NN algorithm has been used, which collects and optimizes the environmental map during running time. To collect the environmental data, the robot uses different sensory modules mounted on the robot body. During training, time controller has been updated many times to generate smooth path that helps the robot to detect the position inside the environment accordingly. To train the robot inside the environment, a neural model depicted in Fig. 3 is used. The model is based on the following equation:

$$Wt_{ij}(n) = Wt_{ij}(n-1) + L_{RN} Y_i^{(k-1)} Y_j^k \tag{12}$$

$$\mathrm{A_o} = \frac{\text{Input data in 'cm' from sensors}}{100\ \text{cm}} \times 180^{\mathrm{o}} \tag{13}$$

where $Y_i^{(k-1)}$ represents the value of the neurons in the output layer of vector $i\,(k-1)$, Y_j^k represents the value of the neurons in the output layer of $j\,(k)$. $Wt_{ij}(n)$ is the weight coefficient of neurons at iteration 'n' to $(n-1)$, respectively. 'L_{RN}' is the learning rate of the network between (0, 0.4). In this analysis, the network has been learned with different examples (Fig. 4). In Fig. 3, input layer, hidden layers 1 and 2, output layer, and actuator have been represented. Weight functions are the inputs of the output layer by which motion in the 'X' and 'Y' directions with theta orientation has been calculated. Finally, velocity of left wheel, velocity of right wheel, and steering angle have been obtained. $S_{1,}S_{2,}\ldots\ldots, S_n$ in the input data sources have shown the two input behaviors from sensory module $u(0)$ to $u(n)$, respectively. Sensory module collected the obstacle distances in the front, right, and left directions of the robot when the robot is navigating toward the target. The minimum collision-free distance of the robot is obtained using Eq. (13) in degrees. These inputs behavior have been implemented to simplify the network architecture. 'S' in the input layer represents the sensor reading for each obstacle which is situated in the target path. The active BNN controller has been used to synchronize the sensory module to calculate both inputs at the run time. The range of the sensor is 10 to 100 cm for detection of obstacles. Based upon these input behaviors, the environment is analyzed.

https://doi.org/10.1017/S0263574719001668 Published online by Cambridge University Press

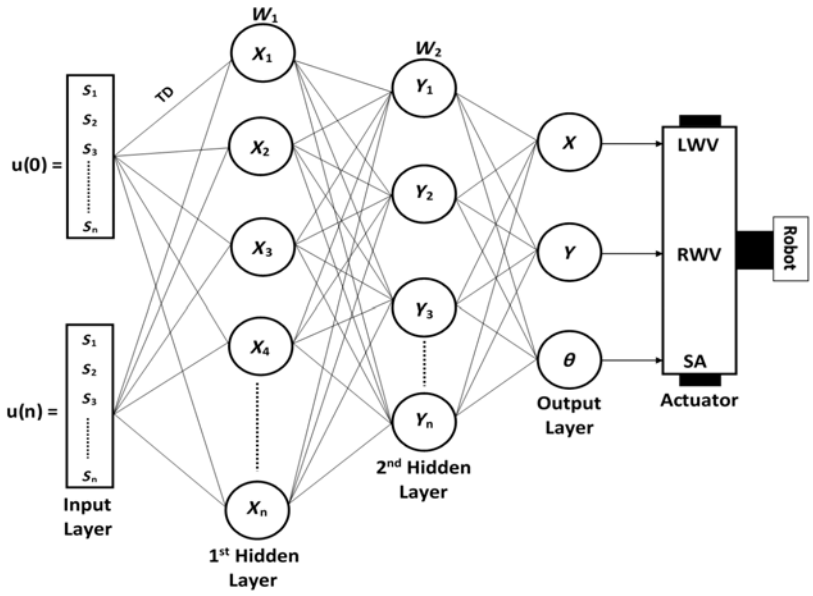

Fig. 3. Operational diagram of BNN in this analysis.

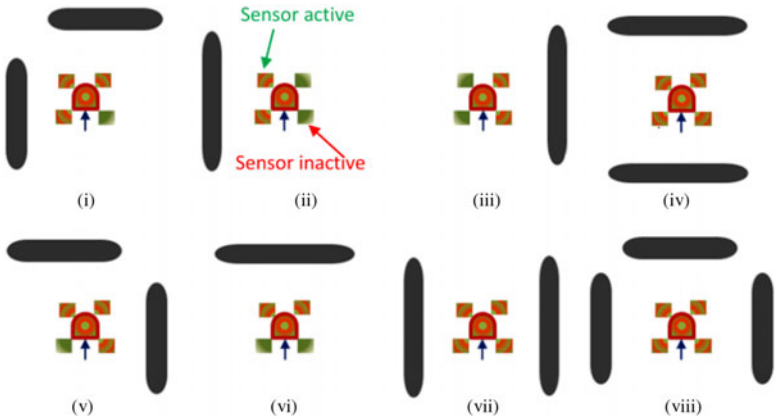

Fig. 4. Different reactive behavior of the mobile robot during training time.

The rules have been generated based upon the input data. Furthermore, the output layer decides the angle toward the target.

Figure 4 shows the different reactive behaviors for the mobile robot during training time. These types of reactive behavior have been used at the time of training. In this figure, eight types (not limited) of reactive behavior have been presented for the training of robot inside the environment. Table I shows the 15 sets of training data for the training. According to the first data from Table I, if left obstacle (LO) is located at a distance of 60 cm, right obstacle (RO) is located at a distance of 80 cm, front obstacle (FO) is located at a distance of 20 cm, and the robot is moving toward the target, then the robot changes its steering angle up to 45° toward the left side to generate the possible path. Similarly, if there is no obstacle within the sensing range (20 cm), then the steering angle is 0° and the robot follows the straight path (Table I rule 6 and 8).

In Table I, some cases represent the negative steering angle and it is due to robot detecting target at the right side. The velocity of both wheels is the same if the steering angle is zero. Similarly, wheel velocity depends upon the direction of obstacles and the target. If a sensor detects any possible obstruction inside the environment at that time, the robot starts the BNN control algorithm. Therefore, the robot has decided the steering angle toward the collision-free path and then updates the target angle according to the position itself. In Fig. 4, it has been depicted that sensor detects the obstacle on its path, and then active signal changes the color ‘green’ (inactive mode) to a combination of ‘red’

https://doi.org/10.1017/S0263574719001668 Published online by Cambridge University Press

Table I. BNN data for the robot training in a disordered environment.

Sl. No.	Obstacles distance in 'cm'			Negotiation angle	Direction of turning
	FO	LO	RO		
1	20	60	80	45	Left
2	100	20	40	55	Left
3	60	80	20	−30	Right
4	80	40	100	0	Straight
5	40	60	60	−60	Right
6	90	40	20	0	Straight
7	100	60	40	10	Left
8	60	90	60	0	Straight
9	80	30	20	−10	Right
10	30	60	90	40	Left
11	80	100	20	20	Left
12	100	20	80	0	Straight
13	90	40	60	−55	Right
14	60	30	80	0	Straight
15	30	80	100	−10	Right

and 'green' (active mode). Curve section represents the front side of the robot, whereas square side represents the rear side (arrow mark) of the robot. In the next section, BNN-based simulation and experiment results have been discussed as well as compared.

4. Results and Discussions

In this section, simulation and real-time experiment results are presented using BNN. The result comparison between simulation and real-time experiment has been depicted graphically in terms of path length and computation time. The time taken in the simulation platform as well as in the experimental platform has been presented in this section. In addition, path length obtained during the navigation from start to the goal position is also taken under discussion for both cases (simulation and experimental). The Khepera II mobile robot (Fig. 9) has been used to conduct real-time experiments. In the upcoming sections, simulation and real-time experiment results have been shown and compared for error calculations.

4.1. Behavior-based NN on MATLAB simulation platform

In this section, simulation experiments using BNN have been presented. First, simulation experiment has been conducted on 300*300 MATLAB R2017a-based arena given in Fig. 5. In this experiment, the mobile robot has successfully completed the navigation and path planning task by avoiding the wall type environment using BNN methodology. In Fig. 5, the robot has avoided the wall using minimum threshold value and reached the target safely. The threshold range is the value by which the robot avoids the obstacles with minimum collision distance or the robot negotiated obstacles with minimum collision distance. At the end of the wall, the robot directly traces the target and follows the starting path to reach the target. During navigation and path planning, the robot has obtained different data from the environment which have been tabulated in Table II. The goal position has been taken as 150 in the '*X*' direction and 100 in the '*Y*' direction. The final distance of the robot from the goal has been obtained as 0.3175 cm (robot stop). In Table II, initially FO, LO, RO and final FOD, LOD, ROD have been given. The number of steps counted during navigation is 234 to reach the target in the environment given by Fig. 5. Initial target angle has been recorded as 8°.

Finally, 1.595 m and 15.53 s, respectively, have been recorded as path length and computation time during the navigation using BNN. Similarly, the second scenario has opted as the simulation experiment given in Fig. 6, and data have been tabulated in Table II. Position of the goal has been taken as 160 in the '*X*' direction and 275 in the '*Y*' direction. The final distance of the robot from the goal has been obtained as 0.0752 cm, and steps have been counted as 244. Initial target angle has been counted as 20.77°. The path length and time taken are calculated as 2.602 m and 12.376 s, respectively. The collision steps encounter during navigation toward the target are recorded as '126'

https://doi.org/10.1017/S0263574719001668 Published online by Cambridge University Press

Table II. Navigational data during simulation experiment using BNN technique.

Scenario	Path length (m)	Time (S)	Initial obstacle distances (cm)			End obstacles distance (cm)			Final robot POS	
			FO	LO	RO	FO	LO	RO	*X*	*Y*
Setup 1 (Fig. 5)	1.595	15.536	303.2	43.85	15.96	177.2	118.0	232.7	150.2	100.1
Setup 2 (Fig. 6)	2.601	12.376	159.6	303.2	16.26	72.04	27.33	293.5	159.9	274.9

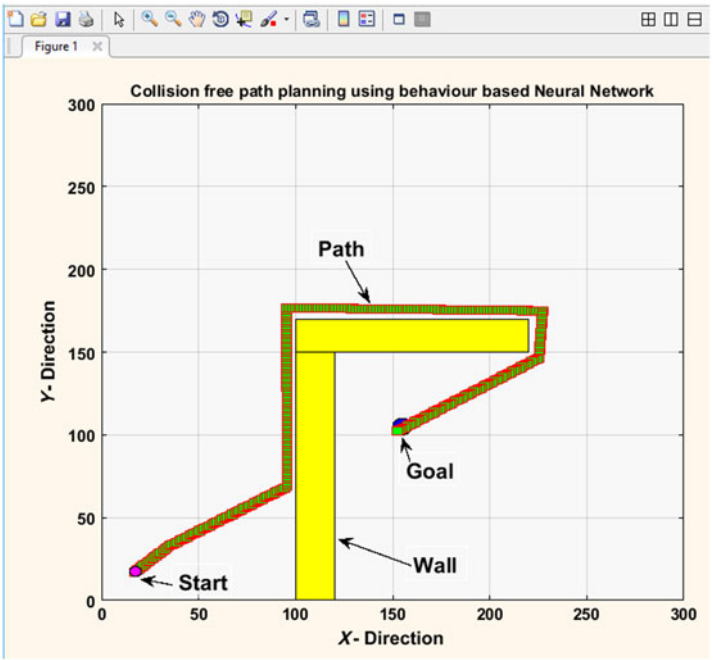

Fig. 5. First simulation experiment using BNN for Collison-free navigation in the regular wall environment.

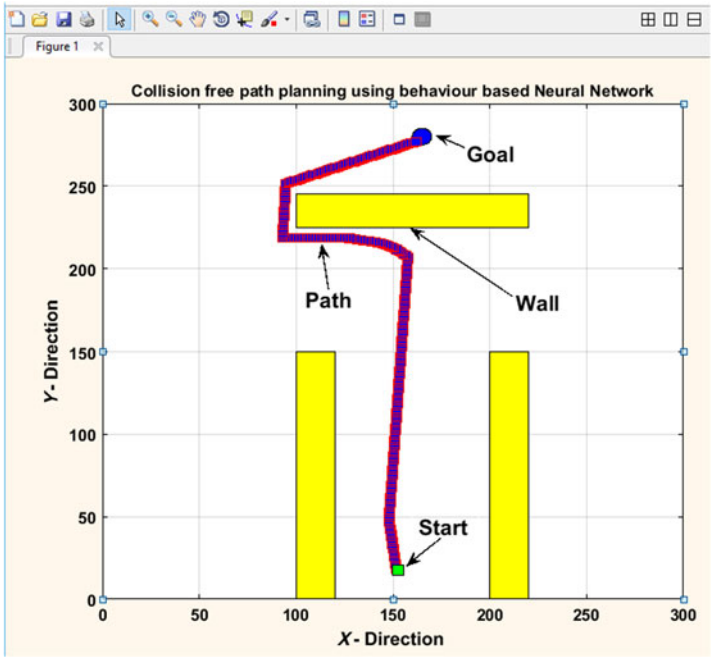

Fig. 6. Second simulation using BNN for Collison-free navigation and wall following behavior.

https://doi.org/10.1017/S0263574719001668 Published online by Cambridge University Press

Table III. Navigational data during real-time experiment using BNN technique in Figs. 7 and 8.

Scenario	Area (*X*, *Y* in cm)	Path length (m)	Time (s)
Setup 1 (Fig. 7)	300, 300	1.76	16.90
Setup 2 (Fig. 8)	300, 300	2.85	13.35

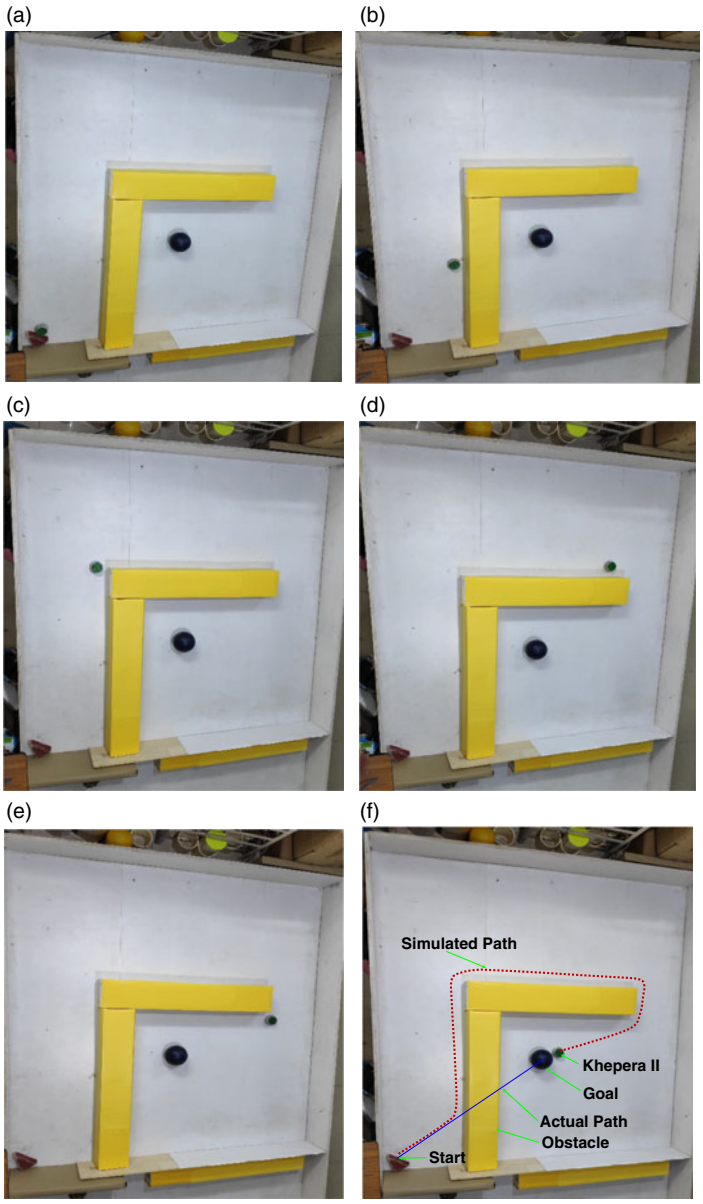

Fig. 7. Real-time experimental result using Khepera II mobile robot in the unknown search space as given in Fig. 5.

and '111' in Figs. 5 and 6, respectively. Finally, improved navigation and path planning behavior has been presented using the BNN control algorithm. In the next section, real-time experimental results have been shown graphical (Figs. 7 and 8) in terms of path length and time taken (Table III).

4.2. Behavior-based NN on real-time experimental platform

In this section, real-time experiments using BNN has been presented. Environments ('*X*' direction = 300 cm, '*Y*' direction = 300 cm) have been created according to the simulation experiment

https://doi.org/10.1017/S0263574719001668 Published online by Cambridge University Press

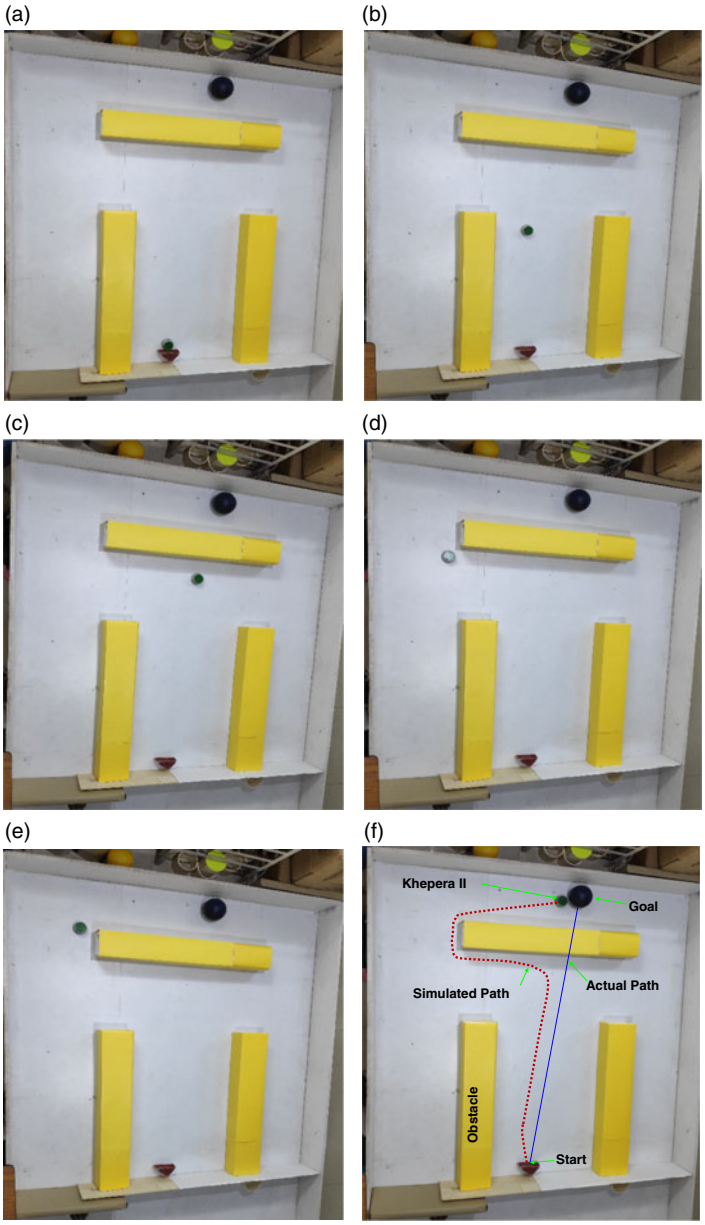

Fig. 8. Khepera II mobile robot achieving the goal in the unknown real-time environment as given in Fig. 6.

environment. The target and robot position are the same, as illustrated in the simulation environment. Finally, real-time experiments have been conducted (Figs. 7 and 8) using the Khepera II mobile robot for validation of simulation (Figs. 5 and 6) results. In the real-time experiment, path length and time taken by the Khepera II mobile robot have been noted and tabularized in Table III. In Fig. 9, specification of the Khepera II mobile robot has been shown. It has 11 number of infrared sensors with a sensing range of 30 cm as well as 5 number of ultrasonic sensors having a range of 4 m. It has a maximum running speed of 0.5 m/s. Inbuilt RAM is 4 kb for DsPIC microcontroller. The comparison between simulation and real-time results has been presented in Table IV in terms of percentage deviation. Percentage deviation in case of path length has been calculated as 9.37 for the first scenario and 8.73 for the second scenario. Similarly, computation time deviation has been recorded as 8.07 for the first scenario and 7.29 for the second scenario. In both of the environments (Figs. 7 and 8), different obstacles state has been presented. In both environments, the robot navigates and achieves the target smoothly. Wall following the event by the robot has been observed in terms of angle deviation, and the deviation in angle is recorded very small in case of a straight wall. Due to the use of

https://doi.org/10.1017/S0263574719001668 Published online by Cambridge University Press

Table IV. Deviation in results for simulation (S) and real-time experiment (E) using BNN.

Sl. No.	Tasks		Path length (m)	Time (s)	Deviation (path)	Deviation (time)
Setup 1	Fig. 5	S	1.595	15.536	9.37	8.07
	Fig. 7	E	1.76	16.90		
Setup 2	Fig. 6	S	2.601	12.376	8.73	7.29
	Fig. 8	E	2.85	13.35		

Processor	DsPIC 30F5011 at 60 MHz
RAM	4 KB on DsPIC
Speed Max	0.02−0.5 m/s
Sensors	11 IR Sensors, 5 Ultrasonic Sensors
Power	Li-polymer (1350 mAh)
Size	D = 130 mm, H = 70 mm
Weight	690 g Approx.
Payload Max	2 Kg Approx.

Fig. 9. Khepera II mobile robot specifications.

BNN methodology, the input functions collected by the robot are more simplified. In addition, the learning capability of a robot is increased due to the use of behavior-based methodology.

In every step of motion, the robot collects obstacle positions and obstacle angles according to target angle as inputs. Therefore, every step of the robot has initialized with obstacle positions and angles, which have been recorded previously. Using BNN methodology, navigation and path planning events have been enhanced. Figure 10 shows the flow chart of BNN-based navigation. In the next section, comparisons with other developed techniques have been presented to show the advancement in BNN-based simulation and real-time navigational results.

In each steps, the robot updates its pose in the local reference coordinates as well as sensors update the obstacles distances according to the robot global reference coordinate. Therefore, the robot plans its path smoothly in the real-time experimental platform using BNN technique. Using the BNN technique, the robot reaction or robot negotiation time with the obstacles is minimized (due to behavior learning technique). Thus, a reactive delay time has been reduced for the narrow corridor, which reduces the overall time taken from start to the goal. During the navigation, the robot encounters collision point very quickly using BNN in both platforms (simulation and experiment). The distance between the robot and the target is updated for every steps according to the global reference frame.

5. Comparison of the Results Between the Proposed and Existing Techniques

In this section, comparison between the proposed technique (BNN) and other developed techniques[26,27] has been illustrated. The proposed BNN technique environments are the same as the other developed techniques. Navigation and path planning events have been successfully completed in this environment using BNN control algorithm. From Figs. 11 and 12, results have been presented in Table V for BNN control algorithm. Graphically, it has been shown in these figures that path created using BNN technique is a more optimized path compared with path developed by fuzzy[26] and neuro-fuzzy.[27] From both of the figures, it has been observed that the mobile robot successfully negotiates with obstacles and safely reaches the target. Figures 11(a) and 12(c) illustrate the more realistic situation when environment is maze and obstacles shape and size are different. Since it is the case of maze environment, then the frequency to switch the steering angle between obstacles and the target has been increased that shows the performance of the proposed control algorithm. MATLAB simulation environments have been explored to design the BNN control algorithm, and the results have been compared with other developed techniques. In Table V, the proposed navigation entities have been tabulated, which are collected during the motion planning of the mobile robot using BNN. In this table, the end pose of the robot in the '*X*' and '*Y*' directions represents the final distance of the robot from '*X*' and '*Y*' axis of the goal.

In Table V, different types of navigational entities have been tabulated for the robot. The target and robot end distance are measured of final distance between robot and the target when robot reaches the

https://doi.org/10.1017/S0263574719001668 Published online by Cambridge University Press

Table V. BNN reactive entities observed during the navigation (Figs. 11(b), (d) and 12(b), (d)).

Sl. No.	Navigation entities		Figure 11(b)	Figure 11(d)	Figure 12(b)	Figure 12(d)
01	End pose in the *X*-direction (cm)		16.9870	2.8024	14.5108	14.4999
02	End pose in the *Y*-direction (cm)		16.9769	3.4064	14.5022	14.4980
03	Target robot end distance (cm)		0.0265	0.0068	0.0111	0.0020
04	Path length, target to robot (cm)		19.4487	15.8937	20.8597	21.0524
05	Total navigational steps		116	96	129	133
06	Collision points encounters		29	36	41	47
07	End target angel (degrees)		60.6153	68.9309	11.5506	87.8951
08	End obstacles distances (cm)	*ROD*	3.5592	3.0726	16.8065	3.5091
09	End obstacles distances (cm)	*LOD*	6.4808	9.9427	3.6069	14.5031
10	End obstacles distances (cm)	*FOD*	3.6495	3.8304	3.7413	3.6844

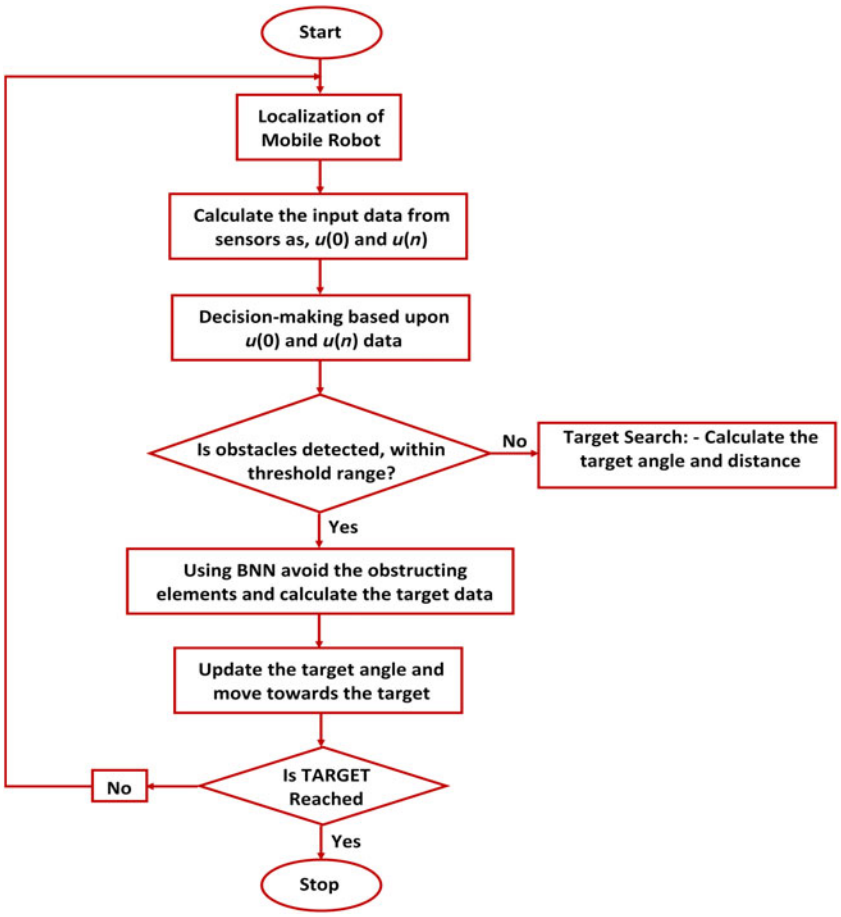

Fig. 10. Obstacle negotiating flow diagram of a mobile robot based on the BNN control technique.

target. The path length recorded by the robot is also tabulated in Table V as well as the target angle at the end has been provided as the end target angle. When the robot reaches the target point at that time, sensor readings concerning obstacle position in the front direction, right direction, and left direction of the robot are observed as the end obstacles distances (Table V). The right, left, and front obstacles distances are tabulated as ROD, LOD, and FOD, respectively. The simulation steps are also provided in this table, which are calculated by the robot during the navigation toward the target. Table VI shows the comparison of results in terms of path length between the proposed technique and other developed techniques. In this comparison, different types of environments have been illustrated in

https://doi.org/10.1017/S0263574719001668 Published online by Cambridge University Press

Table VI. Comparison of results between the proposed technique and other developed techniques.[26,27]

	Figure 11				Figure 12			
	Comparison 1				Comparison 2			
Events	**(a) Fuzzy**	**(b) BNN**	**(c) Fuzzy**	**(d) BNN**	**(a) Neuro-Fuzzy**	**(b) BNN**	**(c) Neuro-Fuzzy**	**(d) BNN**
Path length (m)	23.53	19.44	24.53	15.89	21.33	20.85	23.78	21.05
% Deviation	17.38		35.22		2.25		11.48	

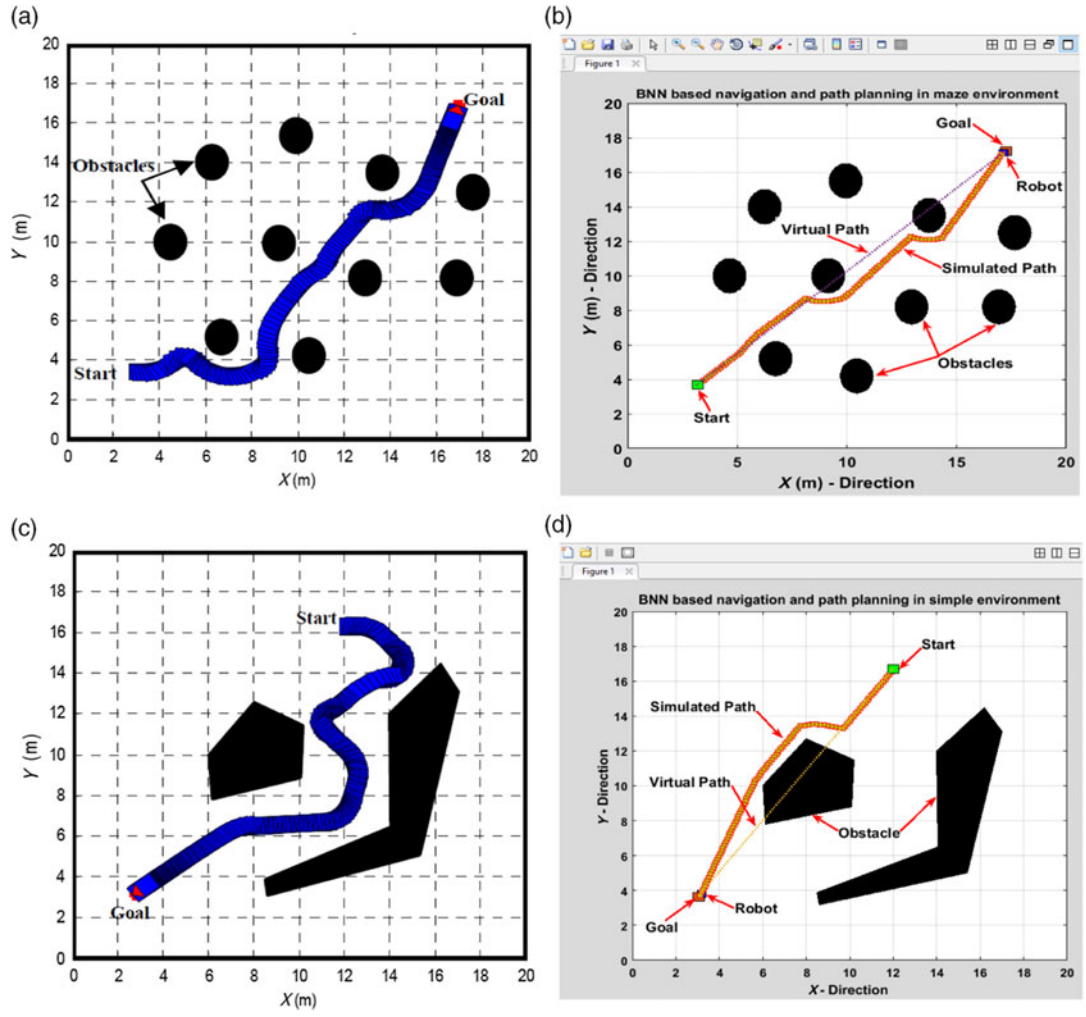

Fig. 11. Comparisons of simulation results: (a) and (c) Cherroun and Boumehraz[26] Fuzzy behavior-based results, (b) and (d) results in the current analysis, and BNN control algorithm.

which robot successfully completed the navigational tasks by avoiding the obstacles. The percentage of deviation is recorded by 17.38%, 35.22%, 2.25%, and 11.48% in terms of navigation path length between the proposed technique and existing techniques.

In Table VI, collision point encounters during the navigation in each environment have been tabulated (Sl. No. 6). It is the point (sum of the total point) at which sensors recognize moving steps with the obstacles and record that steps. Figure 12(d) shows the condition of the maze environment, and the robot records most of the collision steps as '47' in this search space.

https://doi.org/10.1017/S0263574719001668 Published online by Cambridge University Press

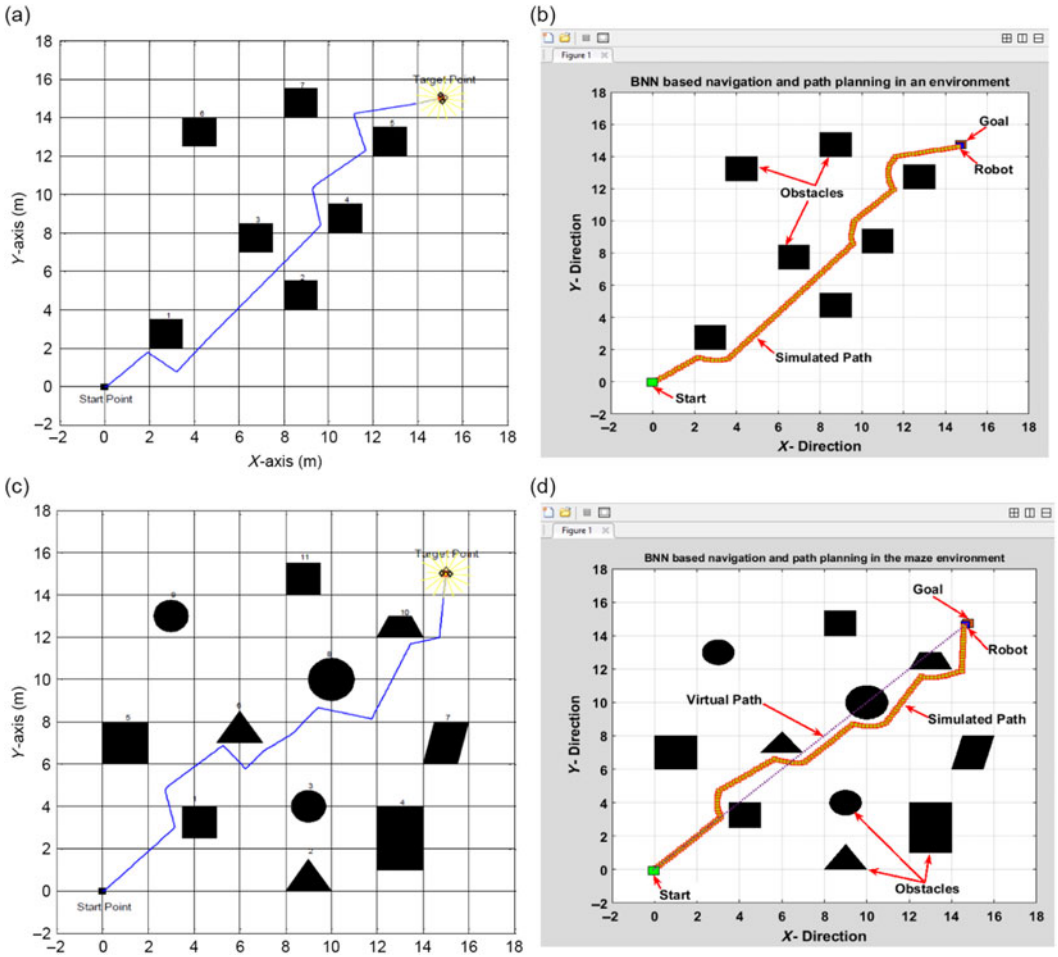

Fig. 12. Comparisons of simulation results: (a) and (c) Mayyahi et al.[27] Neuro-fuzzy results, (b) and (d) result in the current analysis, Behavior-based NN.

6. Conclusions

In this paper, BNN synchronization approach using two behavioral inputs, i.e., obstacle distance $u(0)$ and obstacle angle $u(n)$ according to the target, has been developed. Based on the simulation and the real-time experiments, it is observed that the BNN controller has a better ability to solve the path planning problem in a reasonable time during navigation. To validate the performance of the proposed controller, the results have been presented and compared with the other developed techniques (fuzzy and neuro-fuzzy) in terms of path length. During learning stages, the backpropagation technique has been used to reduce the errors for incoming weights to the neurons that enable the proper learning for the network. In this model, 80% of data have been used in training and learning, and 20% data have been used to test the model. The network used hyperbolic tangent function for hidden node activation. The comparison between the results shows the effectiveness and performance of the BNN control algorithm during the navigation. Navigational effectiveness of the proposed technique has been measured in terms of smoothness of the realistic path, number of collision point detection, path length, and performance time. Using the BNN control algorithm, the robot successfully negotiated the obstacles and reached the target. The current work considers only a single mobile robot in the navigational arena. However, research can be done in future to navigate multiple mobile robots in a common platform using the developed technique.

References

1. R. Fierro and F. L. Lewis, "Control of a nonholonomic mobile robot using neural networks," *IEEE Trans. Neural Networks* **9**(4), 589–600 (1998).
2. M. Khaledyan and M. de Queiroz, "A formation maneuvering controller for multiple non-holonomic robotic vehicles," *Robotica* **37**(1), 189–211 (2019).

https://doi.org/10.1017/S0263574719001668 Published online by Cambridge University Press

3. N. El-Sheimy, K. W. Chiang and A. Noureldin, "The utilization of artificial neural networks for multi-sensor system integration in navigation and positioning instruments," *IEEE Trans. Instrum. Meas.* **55**(5), 1606–1615 (2006).
4. A. Araujo, D. Portugal, M. S. Couceiro and R. P Rocha, "Integrating Arduino-based educational mobile robots in ROS," *J. Intell. Rob. Syst.* **77**(2), 281–298 (2015).
5. L. Marin, M. Valles, A. Soriano, A. Valera and P. Albertos, "Event-based localization in ackermann steering limited resource mobile robots," *IEEE/ASME Trans. Mechatron.* **19**(4), 1171–1182 (2014).
6. L. Wei, C. Cappelle and Y. Ruichek, "Camera/laser/GPS fusion method for vehicle positioning under extended NIS-based sensor validation," *IEEE Trans. Instrum. Meas.* **62**(11), 3110–3122 (2013).
7. M. Mujahed, D. Fischer and B. Mertsching, "Admissible gap navigation: A new collision avoidance approach," *Rob. Auton. Syst.* **103**, 93–110 (2018).
8. M. H. Tanveer, C. T. Recchiuto and A. Sgorbissa, "Analysis of path following and obstacle avoidance for multiple wheeled robots in a shared workspace," *Robotica* **37**(1), 80–108 (2019).
9. D. Gualda, J. Urena, J. C. Garcia, E. Garcia, D. Ruiz and A. Lindo, "Fusion of data from ultrasonic LPS and isolated beacons for improving MR navigation," *IEEE International Conference on Instrumentation and Measurement Technology Conference (IMTC)*, (2014) pp. 1552–1555.
10. F. Khan, A. Alakberi, S. Almaamari and A. R. Beig, "Navigation Algorithm for Autonomous Mobile Robots in Indoor Environments," *Advances in Science and Engineering Technology International Conferences (ASET)*, Abu Dhabi, United Arab Emirates (2018) pp. 1–6.
11. F. A. Da Mota, M. X. Rocha, J. J. Rodrigues, V. H. C. De Albuquerque and A. R. De Alexandria, "Localization and navigation for autonomous mobile robots using petri nets in indoor environments," *IEEE Access* ***6***, 31665–31676 (2018).
12. D. T. Pham and D. R. Parhi, "Navigation of multiple mobile robots using a neural network and a Petri Net model," *Robotica* **21**(1), 79–93 (2003).
13. M. Toth, D. Stojcsics, Z. Domozi and I. Lovas, "Fuzzy Based Indoor Navigation for Mobile Robots," *IEEE 16th International Symposium on Intelligent Systems and Informatics (SISY)*, Subotica, Serbia (2018) pp. 333–338
14. J. Xu, H. Guo and S. Wu, "Indoor Multi-Sensory Self-Supervised Autonomous Mobile Robotic Navigation," *IEEE International Conference on Industrial Internet (ICII)*, Seattle, WA, USA (2018) pp. 119–128
15. S. Cho, J. Park and J. Lee, "A dynamic localization algorithm for a high-speed mobile robot using indoor GPS," *Robotica* **30**(4), 681–690 (2012).
16. J. Lee, J. Lim and J. Lee, "Compensated heading angles for outdoor mobile robots in magnetically disturbed environment," *IEEE Trans. Ind. Electron.* **65**(2), 1408–1419 (2018).
17. J. Simanek, M. Reinstein and V. Kubelka, "Evaluation of the EKF-based estimation architectures for data fusion in mobile robots," *IEEE/ASME Trans. Mechatron.* **20**(2), 985–990 (2015).
18. J. Bai, S. Lian, Z. Liu, K. Wang and D. Liu, "Deep learning based robot for automatically picking up garbage on the grass," *IEEE Trans. Consum. Electron.* **64**(3), 382–389 (2018).
19. D. Speck, C. Dornhege and W. Burgard, "Shakey 2016—How much does it take to redo shakey the robot?," *IEEE Rob. Autom. Lett.* **2**(2), 1203–1209 (2017).
20. L. Kunze, N. Hawes, T. Duckett, M. Hanheide and T. Krajník, "Artificial intelligence for long-term robot autonomy: A survey," *IEEE Rob. Autom. Lett.* **3**(4),4023–4030 (2018).
21. L. Lui and G. S. Sukhatme, "A solution to time-varying markov decision processes," *IEEE Rob. Autom. Lett.* **3**(3), 1631–1638 (2018).
22. A. Pierson, Z. Wang and M. Schwager, "Intercepting rogue robots: An algorithm for capturing multiple evaders with multiple pursuers," *IEEE Rob. Autom. Lett.* **2**(2), 530–537 (2017).
23. H. Zhang, Y. Wang, J. Zheng and J. Yu, "Path planning of industrial robot based on improved RRT algorithm in complex environments," *IEEE Access* **6**, 53296–53306 (2018).
24. X. Luo, S. Li, S. Liu and G. Liu, "An optimal trajectory planning method for path tracking of industrial robots," *Robotica*, **37**(3), 502–520 (2019).
25. D. G. Arango, H. V. Leal, L. M. Hernandez, M. T. S. Pascual and M. H. Sandoval, "Homotopy path planning for terrestrial robots using spherical algorithm," *IEEE Rob. Autom. Sci. Eng.* **15**(2), 567–585 (2018).
26. L. Cherroun and M. Boumehraz, "Fuzzy behavior based navigation approach for mobile robot in unknown environment," *J. Electr. Eng.* **13**(4), 1–8 (2013).
27. A. Al-Mayyahi, W. Wang and P. Birch, "Adaptive neuro-fuzzy technique for autonomous ground vehicle navigation," *Robotics* **3**(4), 349–370 (2014).

https://doi.org/10.1017/S0263574719001668 Published online by Cambridge University Press