Real-time navigational control of mobile robots using an artificial neural network

D R Parhi[1]* and **M K Singh**[2]
[1]Department of Mechanical Engineering, National Institute of Technology Rourkela, Orissa, India
[2]Department of Mechanical Engineering, Government Engineering College Bilaspur, Chhattisgarh, India

The manuscript was received on 27 October 2008 and was accepted after revision for publication on 5 January 2009.

DOI: 10.1243/09544062JMES1410

Abstract: This article deals with the reactive control of an autonomous robot, which moves safely in a crowded real-world unknown environment and reaches a specified target by avoiding static as well as dynamic obstacles. The inputs to the proposed neural controller consist of left, right, and front obstacle distance to its locations and the target angle between a robot and a specified target acquired by an array of sensors. A four-layer neural network has been used to design and develop the neural controller to solve the path and time optimization problem of mobile robots, which deals with cognitive tasks such as learning, adaptation, generalization, and optimization. The back-propagation method is used to train the network. This article analyses the kinematical modelling of mobile robots as well as the design of control systems for the autonomous motion of the robot. Training of the neural net and control performances analysis were carried out in a real experimental set-up. The simulation results are compared with the experimental results and they show very good agreement.

Keywords: evolutionary robotics, artificial neural network, mobile robot, behavioural robotics

1 INTRODUCTION

There is significant interest in autonomous mobile robots, which may be defined as vehicles that are capable of intelligent autonomous navigation. Over the last decade, a great deal of research has reported on machine learning and how it has been applied to help mobile robots optimize their operational capabilities. One of the most important issues in the design and development of an intelligent mobile system is the navigation problem. This consists of the ability of a mobile robot to plan and execute collision-free motions within its environment. However, this environment may be imprecise, vast, dynamical, and either partially structured or non-structured. Robots must be able to understand the structure of this environment [**1**–**5**]. To reach their targets without colliding, robots must be endowed with perception, data processing, recognition, learning, reasoning, interpreting, and decision-making and action capacities.

*Corresponding author: *Department of Mechanical Engineering, NIT Rourkela, C/14, NIT Campus, Rourkela, Orissa 769008, India. email: dayalparhi@yahoo.com*

Service robotics today require synthesizing robust automatic systems able to cope with a complex and dynamic environment [**6**]. To demonstrate this kind of autonomy Muñiz *et al.* [**7**] introduced a neural controller for a mobile robot that learns both forward and inverse odometry of a differential drive robot through unsupervised learning. They introduced an obstacle-avoidance module that is integrated into a neural controller. However, generally, the evolved neural controllers could be fragile in inexperienced environments, especially in real worlds, because the evolutionary optimization processes are executed in idealized simulators. This is known as the gap problem between simulated and real worlds. To overcome this, Kondo [**8**] focused on an evolving on-line learning ability instead of weight parameters in a simulated environment. Based on this, a neuromodulatory neural network model was proposed by them and is utilized as a mobile robot controller. Corradini *et al.* [**9**] used a neural network approach for the solution of the tracking problem of mobile robots. Racz and Dubrawski [**10**] presented a neural network-based approach for mobile robot localization in front of a certain local object. Yang and Meng [**11**] proposed a biologically inspired neural network approach for

real-time collision-free motion planning of mobile robots or robot manipulators in a non-stationary environment. Braganza *et al.* [**12**] described a controller for continuum robots, which utilizes a neural network feed-forward component to compensate the dynamic uncertainties. Research in autonomous multi-robot systems often focuses on mechanisms to enhance the efficiency of the group through some form of cooperation among the individual agents. Moreover, the versatility of a multi-robot system can provide the heterogeneity of structures and functions required to undertake different missions in unknown environmental conditions [**1–3**, **13**].

This article has proposed a neural network-based approach for the solution of the path and time optimization problem of mobile robots. A biologically inspired neural network has been used for real-time collision-free motion planning of mobile robots in an unknown environment. A four-layer perceptron neural network has been used to design the controller. The first layer is used as an input layer, which directly reads from the arrays of sensors of the robot. The neural network consists of two hidden layers, which adjust the weight of the neuron, and an output layer, which provides the heading angle of the robot. The back-propagation method has been used to minimize the error and optimize the path and time of the mobile robot to reach the target.

This article has been organized into five sections. Following the introduction, the kinematics behaviour of the mobile robot is described in section 2. Analysis of the navigation method using neural network architecture is explained in section 3. The simulation results are discussed in section 4. In section 5 experimental results are verified with simulation to demonstrate the superiority of the proposed methodology, and comparison has also been made with other methods [**3**, **14**]. Finally the conclusions are given in section 6.

2 KINEMATICS ANALYSIS OF THE MOBILE ROBOT

The kinematics analysis of the Khepra-III mobile robot has been worked out in this section. The kinematics model of the Khepra-III mobile robot is shown in Fig. 1. It consists of a vehicle chassis with two driving wheels mounted on the same axis and a front point sliding support. The two driving wheels are independently driven by two actuators to achieve the motion and orientation. Both wheels have the same diameter denoted by $2r$ (Fig. 2). The two driving wheels are separated by distance W. The centre of gravity (COG) of the mobile robot is located at point 'C'. Point 'P' is located at the intersection of a straight line passing through the middle of the vehicle and a line passing through the axis of the two wheels. The distance

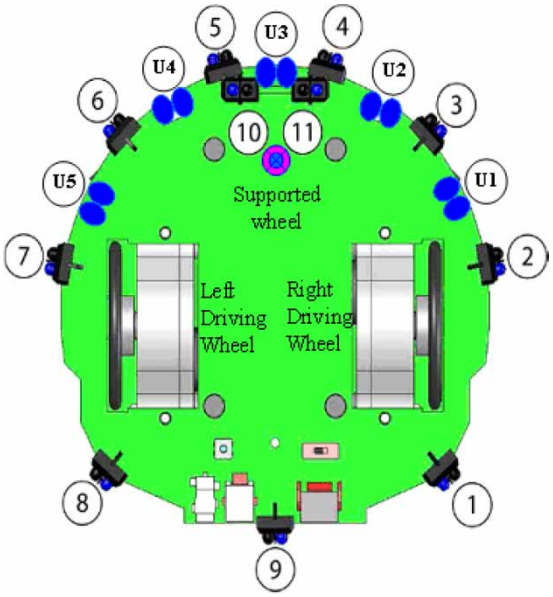

Fig. 1 Chassis of the Khepra-III robot

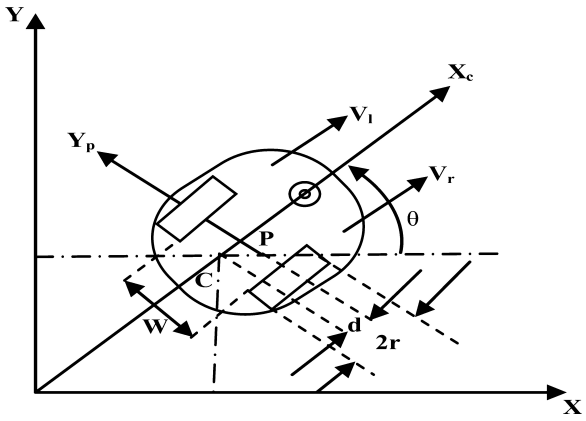

Fig. 2 Kinematics of the mobile robot

between points P and C is d. A motion controller based on a neural network technique has been proposed for navigation of the mobile robot. The main component in the motion controller is the low-level inverse neural controller, which controls the dynamics of the mobile robot.

The kinematics of the differential drive mobile robot is based on the assumption of pure rolling and there is no slip between the wheel and surface

$$v_t = \frac{1}{2}(v_r + v_l) \tag{1}$$

$$\omega_t = \frac{1}{w}(v_r - v_l) \tag{2}$$

$$v_r = r\omega_r \quad \text{and} \quad v_l = r\omega_l \tag{3}$$

where v is the linear velocity and ω is the angular velocity of the vehicle. Superscript r, l, and t stand for right wheel, left wheel, and tangential (with respect to its COG point 'C' measured in a right wheel), respectively.

The position of the robot in the global coordinate frame $[O\ X\ Y]$ is represented by the vector notation as

$$\mathbf{q} = [X_c \ \ Y_p \ \ \theta]^T \tag{4}$$

where X_c and Y_p are the coordinates of point P in the global coordinate frame (Fig. 2). The variable θ is the orientation of the local coordination of the local coordinate frame $[P\ X_c\ Y_p]$ attached on the robot platform measured from the horizontal axis. Three generalized coordinates can describe the configuration of the robot as equation (4).

The mobile robot system considered here is a rigid body and the wheels are pure rolling and no slippage. This states that the robot can only move in the direction normal to the axis of the driving wheels. Therefore, the component of the velocity of the contact point with the ground, orthogonal to the plane of the wheel, is zero [**15**, **16**], i.e.

$$[\dot{y}_p \cos\theta - \dot{x}_c \sin\theta - d\dot{\theta}] = 0 \tag{5}$$

All kinematics constraints are independent of time, and can be expressed as

$$\mathbf{A}^T(\mathbf{q})\dot{q} = 0 \tag{6}$$

where $\mathbf{A}(\mathbf{q})$ is the input transformation matrix associated with the constraints

$$\mathbf{C}^T\mathbf{A}(\mathbf{q}) = 0 \tag{7}$$

where $\mathbf{C}(\mathbf{q})$ is the full rank matrix formed by a set of smooth and linearly independent vector fields spanning the null space of $\mathbf{A}^T(\mathbf{q})$.

From equations (6) and (7) it is possible to find an auxiliary vector time function $\boldsymbol{V}(t)$ for all time t

$$\dot{q} = \mathbf{C}(\mathbf{q})\boldsymbol{V}(t) \tag{8}$$

The constraint matrix in equation (6) for a mobile robot is given by

$$\mathbf{A}^T(\mathbf{q}) = [-\sin\theta \ \ \cos\theta \ \ -d] \tag{9}$$

The $\mathbf{C}(\mathbf{q})$ matrix is given by

$$\mathbf{C}(\mathbf{q}) = \begin{bmatrix} \cos\theta & -d\sin\theta \\ \sin\theta & d\cos\theta \\ 0 & 1 \end{bmatrix} \tag{10}$$

and

$$\boldsymbol{V}(t) = [v \ \ \omega]^T \tag{11}$$

where v is the linear velocity of point 'p' along the robot axis and ω is the angular velocity.

Therefore, the kinematics equation in (8) can be described as

$$\dot{q} = \begin{bmatrix} \dot{x}_c \\ \dot{y}_p \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & -d\sin\theta \\ \sin\theta & d\cos\theta \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{12}$$

Equation (12) is called the steering system of the vehicle. The control problem is to find a suitable control law so that the system can track desired reference trajectories. The control laws are designed to produce suitable left and right wheel velocities for driving the mobile robot to follow required path trajectories.

3 ANALYSIS OF A NEURAL NETWORK FOR NAVIGATION

Artificial neural networks consist of a set of simple, densely interconnected processing units. These units transform signals in a non-linear way. Neural networks are non-parametric estimators that can fit smooth functions based on input–output examples [**17**]. The neural network designed in this article is a four-layer perceptron. The number of layers is set empirically to facilitate the training. The input layer has four neurons, three for receiving the values of the distances from obstacles (i.e. in front and to the left and right of the robot) and one for the target bearing. If no target is detected, the input to the fourth neuron is set to 'zero'. The output layer has a single neuron, which produces the steering angle to control the direction of movement of the robot. The first hidden layer has ten neurons and the second hidden layer has three neurons. These numbers of hidden neurons were also found empirically. Figure 3 depicts the neural network with its input and output signals.

The neural network is trained to navigate by presenting it with 200 patterns representing typical scenarios,

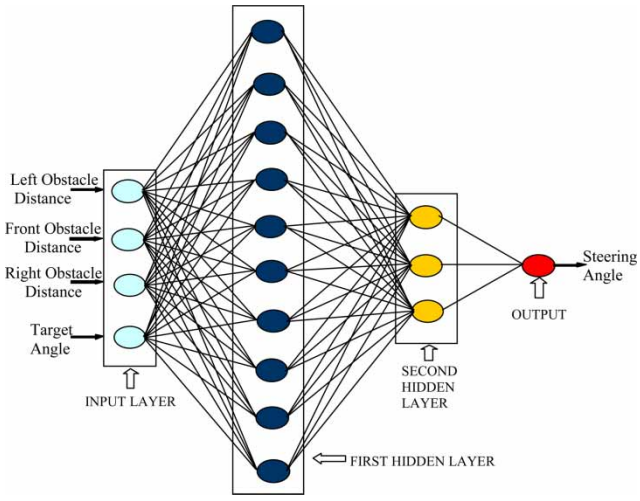

Fig. 3 Four-layer neural network for robot navigation

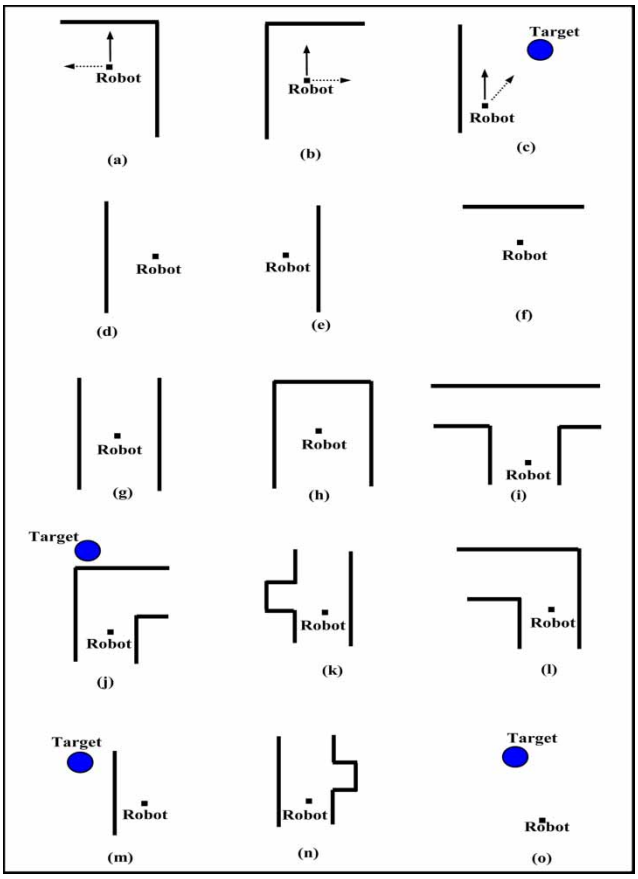

Fig. 4 Example training patterns

some of which are depicted in Fig. 4. For example, Fig. 4(a) shows a robot advancing towards an obstacle, another obstacle being on its right hand side. There are no obstacles to the left of the robot and no target within sight. The neural network is trained to output a command for the robot to steer towards its left.

During training and during normal operation, the input patterns fed to the neural network comprise the following components

$$y_1^{\{1\}} = \text{left obstacle distance from the robot} \tag{13a}$$

$$y_2^{\{1\}} = \text{front obstacle distance from the robot} \tag{13b}$$

$$y_3^{\{1\}} = \text{right obstacle distance from the robot} \tag{13c}$$

$$y_4^{\{1\}} = \text{target bearing} \tag{13d}$$

These input values are distributed to the hidden neurons, which generate outputs given by [**17**]

$$y_j^{\{\text{lay}\}} = f(V_j^{\{\text{lay}\}}) \tag{14}$$

where

$$V_j^{\{\text{lay}\}} = \sum_i W_{ji}^{\{\text{lay}\}} y_i^{\{\text{lay}-1\}} \tag{15}$$

lay = layer number (2 or 3), j = label for jth neuron in hidden layer 'lay', i = label for ith neuron in hidden layer 'lay-1', $W_{ji}^{\{\text{lay}\}}$ = weight of the connection from

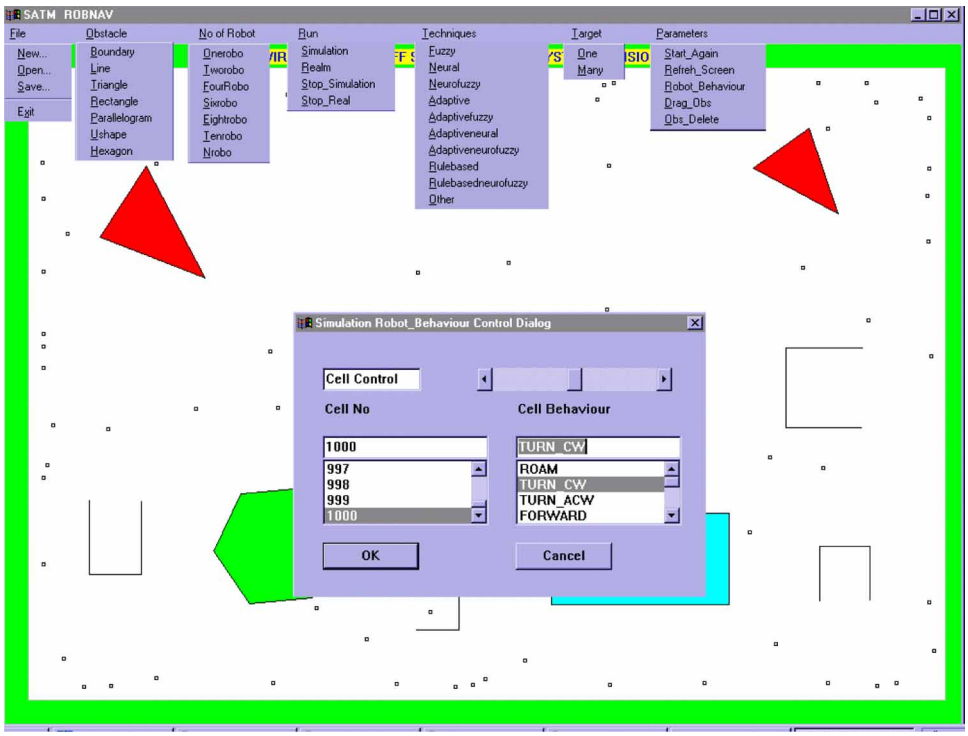

Fig. 5 Robot navigation software package (ROBNAV)

Table 1 Reactive behaviours adopted by a mobile robot during navigation

Type of behaviour	Description of the behaviour	Implementations
Obstacle avoidance	(a) The mobile robot detects (by sensory information) any obstacle in the front, left, or right side. This behaviour is required to avoid collision with obstacles (Fig. 6) (b) When the acquired information from the sensors shows the presence of obstacles to the front, left, and right side of the robot, the robot reverses its movement (Fig. 8)	The robot reduces the speed and sets the steering angle accordingly The robot stops and takes counterclockwise rotation for both the left and right wheel at the same speed (i.e. reverse direction)
Target seeking	When the acquired information from the sensors shows that there are no obstacles around the robot, its main reactive behaviour is to seek the target. This behaviour is required in locating the target (Figs 6 to 8)	The robot mainly adjusts its motion direction and quickly moves towards the target
Wall following	The mobile robot detects an obstacle in the front while it is moving towards the target and also has the wall to the left or right side. The robot has to follow the wall to reach the target (Fig. 7)	The robot adjusts the speed and sets the heading angle 90° with the wall so that it aligns with the wall and moves along the wall. The robot automatically makes a turn to align itself along the wall and moves in parallel with the wall to reach the target

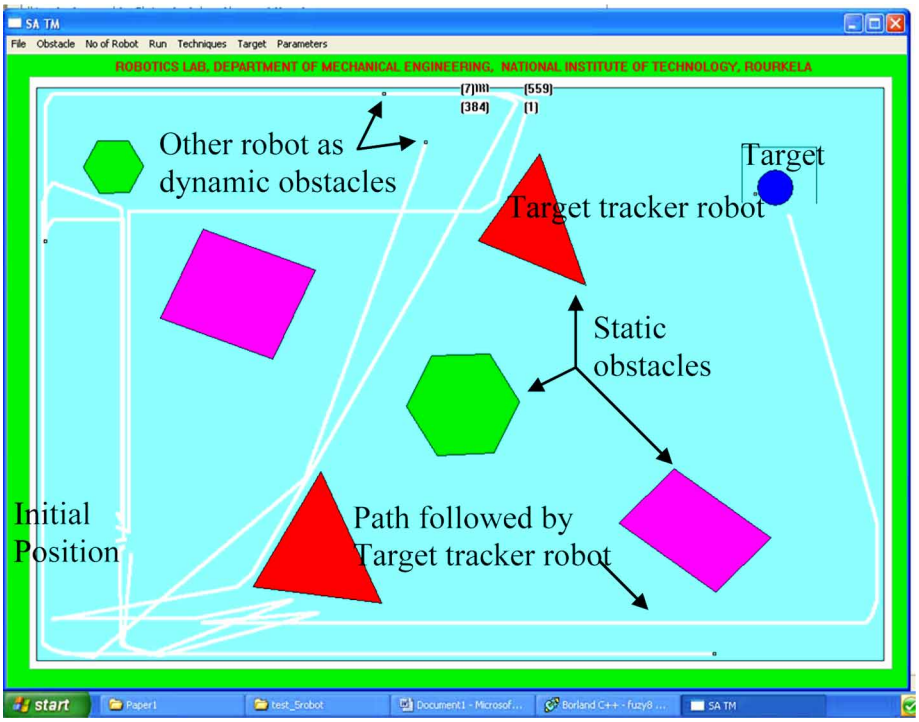

Fig. 6 Static as well as dynamic obstacle-avoidance behaviour

neuron i in layer 'lay-1' to neuron j in layer 'lay', and $f(\cdot)$ = activation function, chosen in this work as the hyperbolic tangent function

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{16}$$

During training, the network output θ_{actual} may differ from the desired output θ_{desired} as specified in the training pattern presented to the network. A measure of the performance of the network is the instantaneous sum-squared difference between θ_{desired} and θ_{actual} for the set of presented training patterns

$$\text{Err} = \frac{1}{2} \sum_{\substack{\text{all training} \\ \text{patterns}}} (\theta_{\text{desired}} - \theta_{\text{actual}})^2 \tag{17}$$

The error back-propagation method is employed to train the network [**17**]. This method requires the computation of local error gradients in order to determine appropriate weight corrections to reduce Err. For the output layer, the error gradient $\delta^{\{4\}}$ is

$$\delta^{\{4\}} = f'(V_1^{\{4\}})(\theta_{\text{desired}} - \theta_{\text{actual}}) \tag{18}$$

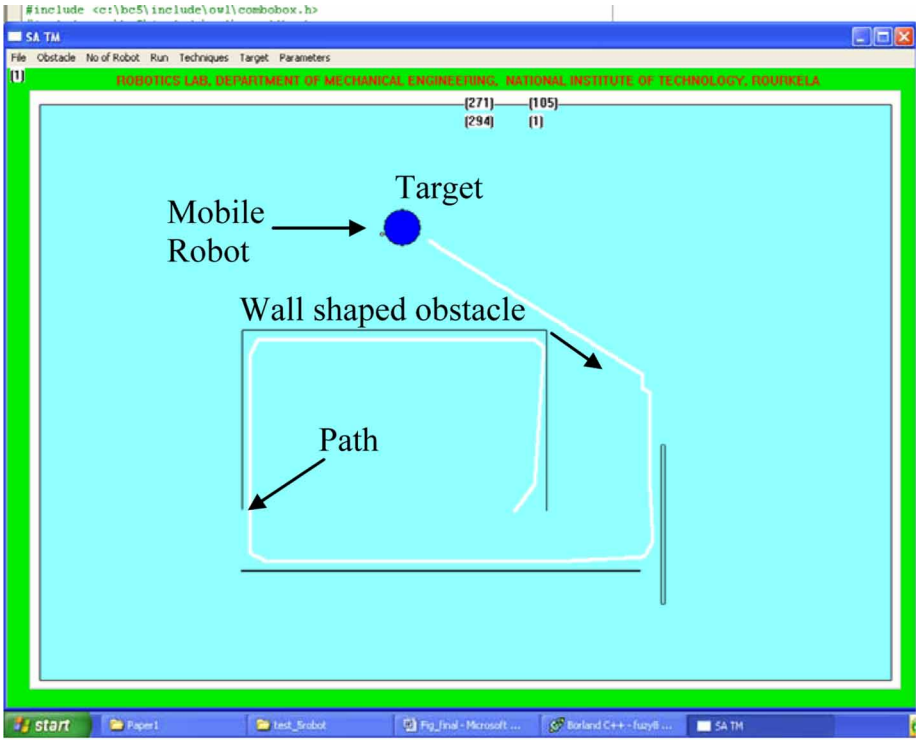

Fig. 7 Robot escaping from U-shaped wall

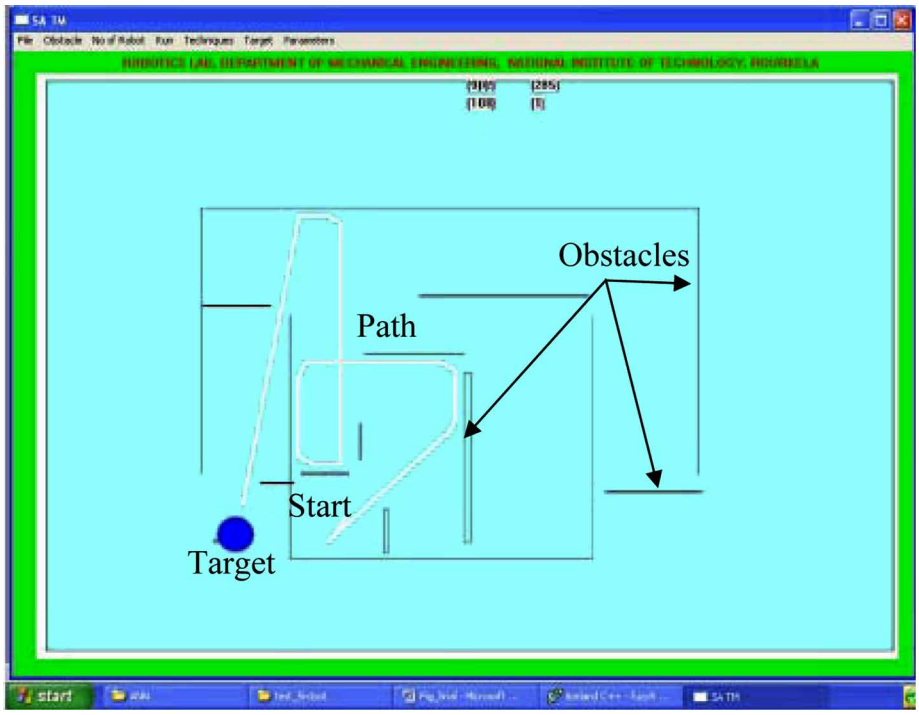

Fig. 8 Robot escaping from dead-end obstacle

The local gradient for neurons in hidden layer {lay} is given by

$$\delta_j^{\{\text{lay}\}} = f'(V_j^{\{\text{lay}\}}) \left(\sum_k \delta_k^{\{\text{lay}+1\}} W_{kj}^{\{\text{lay}+1\}} \right) \tag{19}$$

The synaptic weights are updated according to the expressions

$$W_{ji}(t+1) = W_{ji}(t) + \Delta W_{ji}(t+1) \tag{20}$$

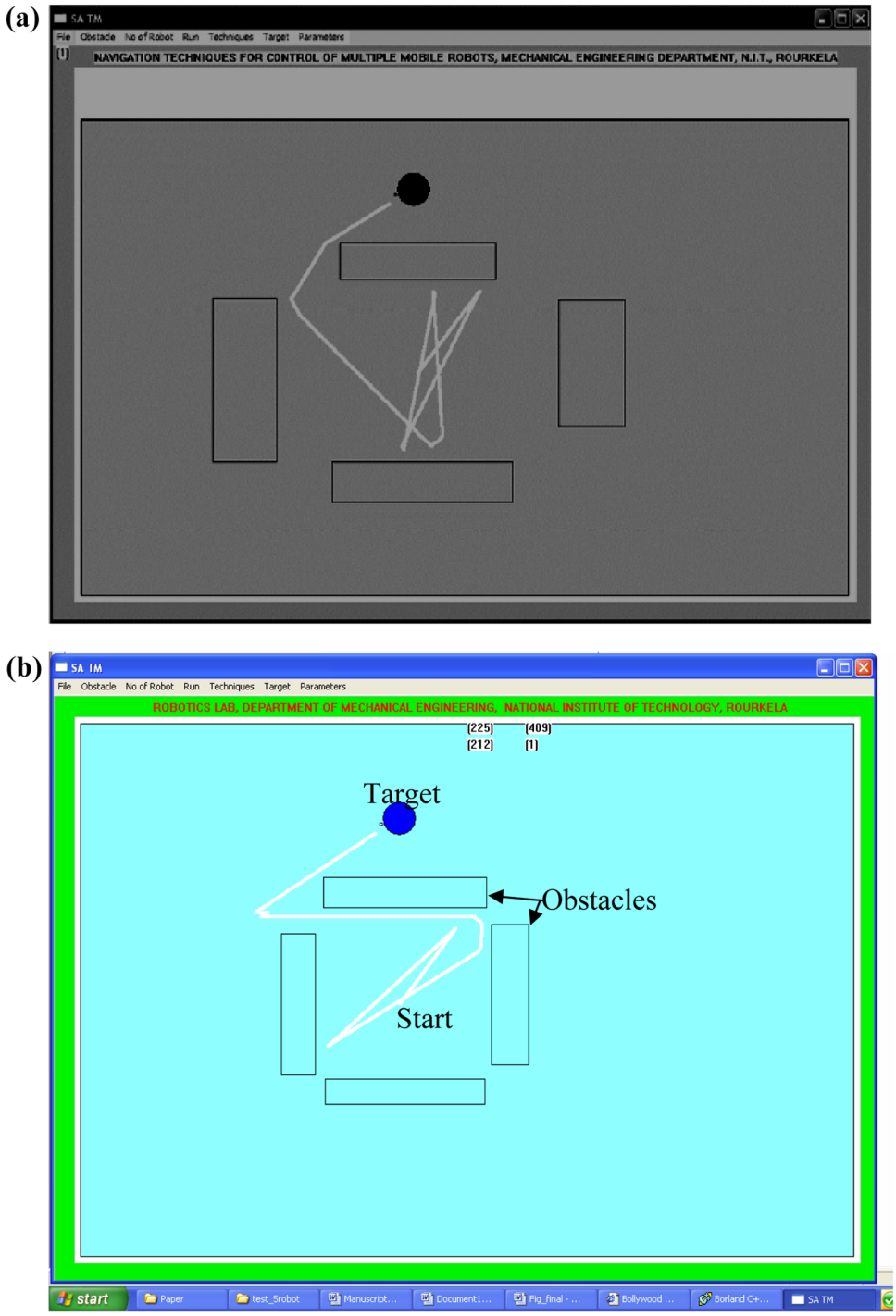

Fig. 9 Result comparisons with Pradhan *et al.* (a) Navigation path of a mobile robot using a fuzzy controller by Pradhan *et al.* and (b) navigation path of a mobile robot using a proposed neural controller

and

$$\Delta W_{ji}(t+1) = \alpha \Delta W_{ji}(t) + \eta \delta_j^{\{\text{lay}\}} y_i^{\{\text{lay}-1\}} \tag{21}$$

where $\alpha =$ momentum coefficient (chosen empirically as 0.2 in this work), $\eta =$ learning rate (chosen empirically as 0.35 in this work), $t =$ iteration number, each iteration consisting of the presentation of a training pattern and correction of the weights.

The final output from the neural network is

$$\theta_{\text{actual}} = f(V_1^{\{4\}}) \tag{22}$$

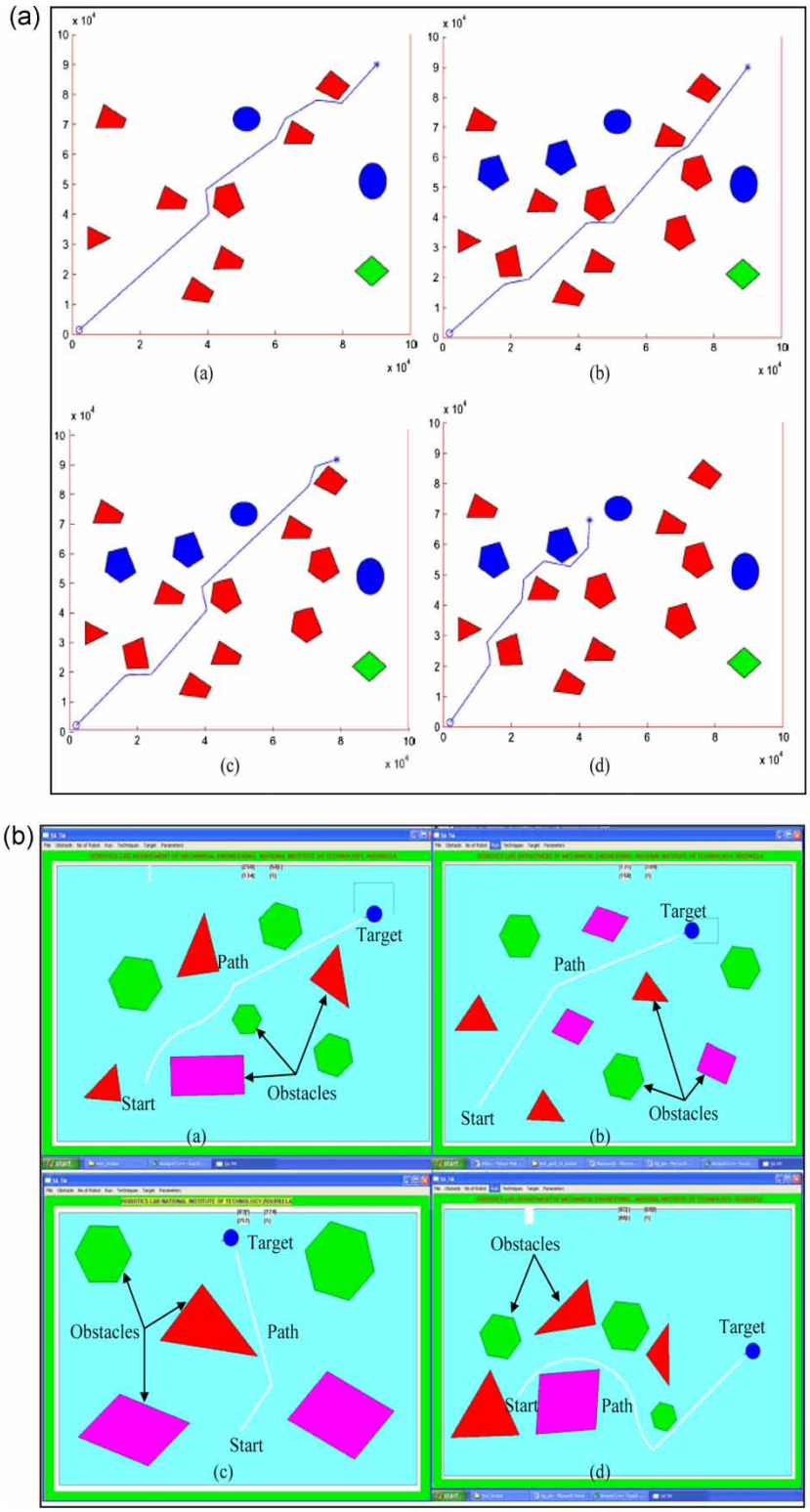

Fig. 10 Result comparisons with Ray *et al.* (a) Navigation of a mobile robot in unknown environment by Ray *et al.* and (b) navigation of a mobile robot in unknown environment using a developed controller

where

$$V_1^{\{4\}} = \sum_i W_{1i}^{\{4\}} y_i^{\{3\}} \tag{23}$$

It should be noted that learning can take place continuously even during normal target seeking behaviour. This enables the neural controller to adopt the changes in the robot's path while moving towards the target.

The proposed neural controller and kinematics give the steering angle from wheel velocities based on the environmental conditions.

4 SIMULATION RESULTS

Simulations are conducted with the ROBNAV software being developed in the laboratory using C++ [**18**], as with C++ it is easier to talk with hardware and access physical memory. Figure 5 shows a typical screen of the software. It can be noted that, in addition to the neural network-based navigation, the software also allows other navigation control. The techniques menu in the software enables the user to select techniques to control the navigation of robots. This menu also allows a user to embed external controller code (coded by third party). The real windows environment (e.g. obstacle, target, speed, path length, time, etc.) of the software has been coded for generic application. For efficient real-time navigation of a mobile robot in a real world environment, the reactive behaviours adopted by the robot during navigation are shown in Table 1. To demonstrate the effectiveness and robustness of the proposed method, simulation results on mobile robot navigation in various environments are exhibited.

Obstacle avoidance behaviour is activated when the readings from any sensor are less than the minimum threshold values. This is how the robot determines if an object is close enough for a collision. When an object is detected too close to the robot, it avoids the collision by moving away from it in the opposite direction. Collision avoidance is the highest priority and, therefore, it can override other behaviours; in this case, its main reactive behaviour is decelerating for static as well as dynamic obstacle avoidance as shown in Fig. 6.

The wall following behaviour mode was adopted when the mobile robot detects an obstacle in the front while it is moving towards the target along the left or right side of the wall. The mobile robot may turn left or right due to the presence of an obstacle in the front. Another special condition appears as the mobile robot detects an obstacle in the front while the target tracking control mode is in operation. In this case, the fixed wall following behaviour should be performed on a priority basis. That is, the mobile robot must rotate clockwise or counterclockwise such that it can align and move along the wall, when the robot is moving to a specified target through a narrow channel or escaping from a U-shaped obstacle (Fig. 7). In the absence of wall following behaviour, the robot is incapable of reaching the goal position when it encounters U-shaped or dead-end obstacles in its path. In such a situation the robot should keep on heading towards the goal position. But when it moves towards the goal position, the robot also comes closer to the obstacles. Any obstacle-avoidance behaviour except wall following behaviour would make the robot divert from its goal position. Figures 7 and 8 exhibit a wall following behaviour along with target seeking.

When the acquired information from the sensors shows that there are no obstacles around the robot, its main reactive behaviour is target steer. A neural controller mainly adjusts robot motion direction and steers the robot towards the target if there are no obstacles around the robot as shown in Fig. 6. In the proposed control strategy, reactive behaviours are formulated and trained by a neural network.

The results from the proposed neural controller were compared with the results from the fuzzy controller by Pradhan *et al.* [**3**] (Fig. 9). Also, a comparison was made between the results obtained by Ray *et al.* [**14**] for navigation of a mobile robot in an unknown environment and the results from the current developed controller (Fig. 10). They show good agreement.

5 EXPERIMENTAL RESULTS

Experimental results were obtained using the Khepra-III mobile robot. The chassis of the robot measures 130 × 70 mm ($D \times H$), weight 690 g, pay load ≈2000 g fitted DsPIC 30F5011 at 60 MHz processor with 4 KB on DsPIC and 64 MB KoreBot RAM. The robot has two DC brushed servo motors with incremental encoders (roughly 22 pulses per mm of robot motion), and nine infrared proximity and ambient light sensors with up to 25 cm range, two infrared ground proximity sensors for line following applications, and five ultrasonic sensors with range 0.2–4 m. The speed range of the robot is 0–0.5 m/s, with a power adapter lithium–polymer battery pack (1400 mAh) as shown in Fig. 11. The assumptions for real navigation control of a mobile robot are as follows:

(a) a mobile robot moves on a plane surface (on lab-specified floor area);

Fig. 11 Khepra-III mobile robot developed by K-team

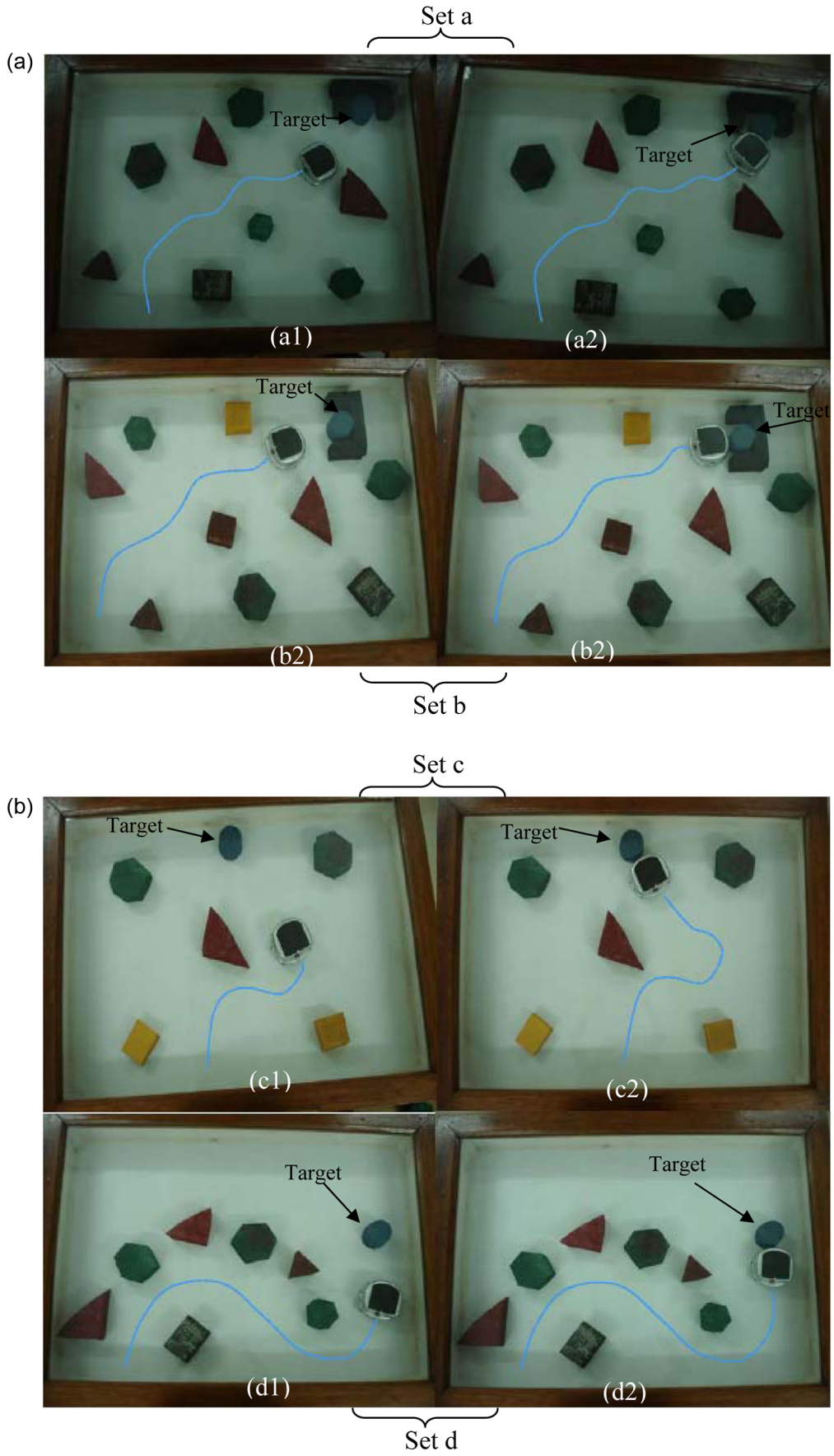

Fig. 12 (a) and (b): experimental results during target seeking by the mobile robot

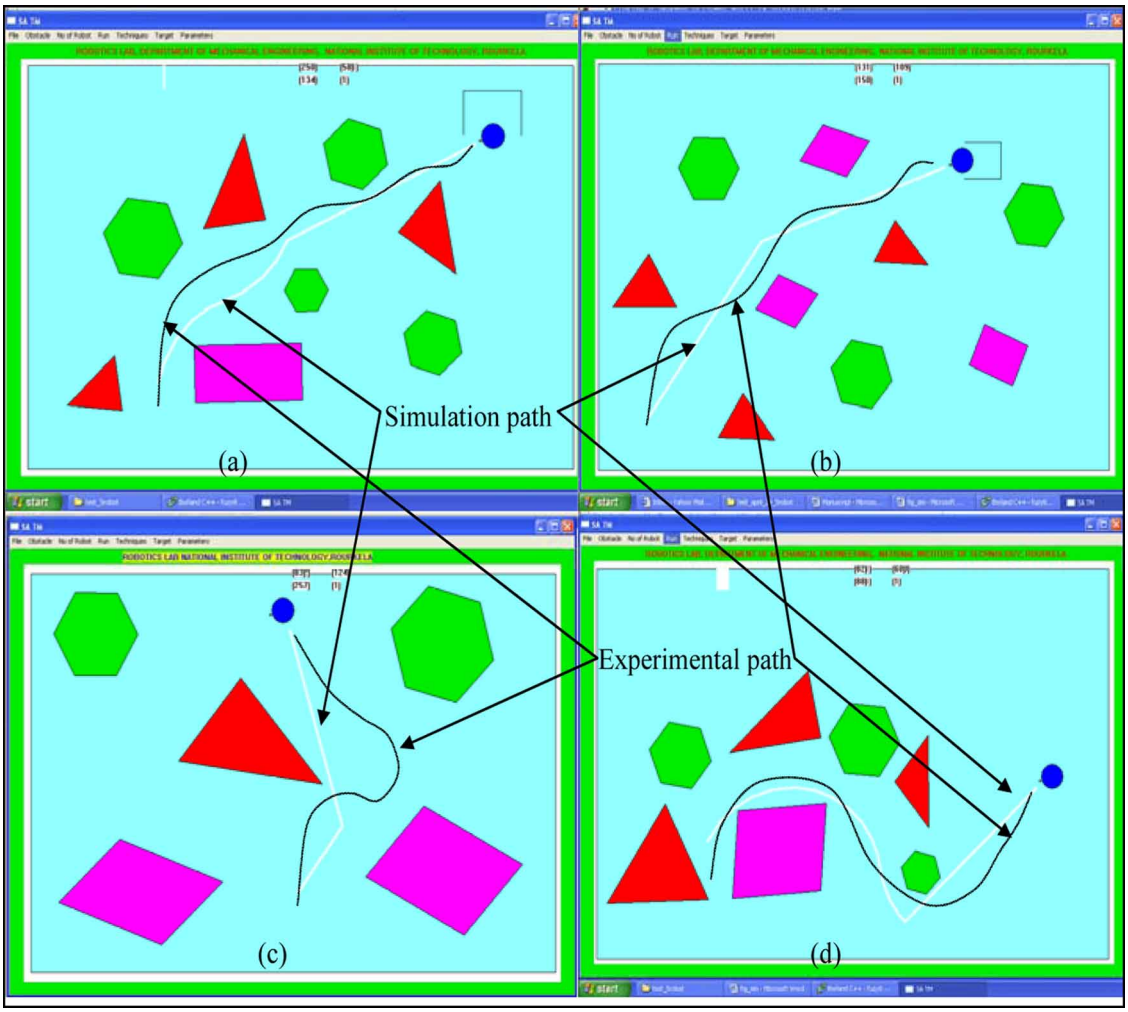

Fig. 13 Comparison of experimental results with simulation results

Table 2 Time taken by robots in simulation and experiment to reach targets

S. no.	Average of nine experiments in each environment	Path length (in pixels)	Time during simulation (s)	Time during experiment (s)
1	For first environment scenario Fig. 12(a) (set a) and Fig. 13(a)	221	23	26
2	For second environment scenario Fig. 12(a) (set b) and Fig. 13(b)	206	22	25
3	For third environment scenario Fig. 12(b) (set c) and Fig. 13(c)	175	18	21
4	For fourth environment scenario Fig. 12(b) (set d) and Fig. 13(d)	234	25	27

(b) the wheel of a mobile robot rolls on the floor without any translational slip.

The experimental paths followed by mobile robots to reach the target are traced. From the neural controller (inputs: left, front, right obstacle distances, and heading angle) after learning, training, and testing, robots get the left and right wheel velocities, which subsequently give the new steering angles. The paths traced by the robots are marked on the floor by a pen (fixed to the front of the robots) as they move as shown in Fig. 12. The experimentally obtained paths follow closely those traced by the robots during simulation (Fig. 13). During experiments, it has been found that the experimental path length and time are more than the simulation path length and time. This is due to the presence of various errors (e.g. signal transmission error, obstacle/target tracking error, friction in rotating elements, slippage between floor and wheels, friction between supported point and floor, etc.).

From these figures, it can be seen that the robots indeed avoid obstacles and reach the targets. Table 2

shows the times taken by the robots in simulations and in experimental tests to find the targets. The figures given are the averages of three experiments on each environmental scenario conducted in the laboratory. These robotic behaviours are verified in simulation and experimental mode (Figs 12 and 13). It is observed that the robots are able to reach the targets efficiently during simulation and experiment. The fuzzy logic controller proposed by Pradhan *et al.* [**3**] has been examined and compared with the proposed neural controller in a similar navigational environment. It has been found that the neural controller gives a more optimized path than the fuzzy controller (the total path length using a fuzzy controller by Pradhan *et al.* [**3**] is 13.7 m and the time taken is 14.67 s to reach the target, whereas the total path length using a proposed neural controller is 12.0 m and the time taken is 12.84 s). In addition, a neural controller requires less computing time and computing memory than a fuzzy controller. It has been observed that the robot controlled through neural control has better performance than the fuzzy controller in terms of positioning accuracy and collision avoidance.

6 CONCLUSIONS

From the kinematics, simulations, and experimental analysis, the following conclusions are drawn.

The effectiveness of the developed controller has been verified in both simulation and real mode and they are in good agreement. The simulation results are also compared with the results obtained from the other investigation and they are in very good agreement. The back-propagation neural network technique has been used for making the controller. Software has been developed using C++ to obtain the simulation results. The developed neural controller has the following salient features:

(a) avoids static as well as dynamic obstacles along the path;
(b) the robot rapidly recognizes its surroundings, which provides sufficient information for path optimization during navigation;
(c) behaviours such as obstacle avoidance, wall following, and target seeking are integrated in the current controller to obtain an efficient navigational controller;
(d) the developed neural controller is more efficient than the fuzzy logic controller (Pradhan *et al.*);
(e) the proposed neural controller is a simple but efficient tool for mobile robot navigation, especially in a dynamic environment;
(f) training patterns of each network can be generated by simulation rather than by experiment, saving considerable time and effort.

Some features of the intelligent controller cannot be added by using a single technique like fuzzy logic or neural network technique. Certainly, these two fields can be integrated into a new emerging technology called neuro-fuzzy or fuzzy-neuro, which combines the benefits of each field (i.e. perception, cognition, and motion control). In future analysis a hybrid controller can be designed for more efficient navigation of the mobile robots.

REFERENCES

1 **Nelson, A. L., Grant, E.,** and **Henderson, T. C.** Evolution of neural controllers for competitive game playing with teams of mobile robots. *Robot. Auton. Syst.*, 2004, **46**, 135–150.
2 **Parhi, D. R., Pradhan, S. K., Panda, A. K.,** and **Behra, R. K.** The stable and precise motion control for multiple mobile robots. *Appl. Soft Comput.*, 2009, **9**, 477–487.
3 **Pradhan, S. K., Parhi, D. R.,** and **Panda, A. K.** Fuzzy logic techniques for navigation of several mobile robots. *Appl. Soft Comput.*, 2009, **9**, 290–304.
4 **Wolf, D. F.** and **Sukhatme, G. S.** Semantic mapping using mobile robots. *IEEE Trans. Robot.*, 2008, **24**(2), 245–258.
5 **Gregor, K.** and **Igor, S.** Tracking-error model-based predictive control for mobile robots in real time. *Robot. Auton. Syst.*, 2007, **55**, 460–469.
6 **Folgheraiter, M., Gini, G., Nava, A.,** and **Mottola, N.** A BioInspired neural controller for a mobile robot. In Proceedings of the IEEE International Conference on *Robotics and biomimetics*, Kunming, China, 17–20 December 2006, pp. 1646–1651.
7 **Muñiz, F., Zalama, E., Gaudiano, P.,** and **López Coronado, J.** Neural controller for a mobile robot in a non-stationary environment. In Proceedings of the 2nd IFAC Conference on *Intelligent autonomous vehicles*, Helsinki, 12–14 June 1995, pp. 1–7.
8 **Kondo, T.** Evolutionary design and behavior analysis of neuromodulatory neural networks for mobile robots control. *Appl. Soft Comput.*, 2007, **7**, 189–202.
9 **Corradini, M. L., Ippoliti, G.,** and **Longhi, S.** Neural networks based control of mobile robots: development and experimental validation. *J. Robot. Syst.*, 2003, **20**(10), 587–600.
10 **Racz, J.** and **Dubrawski, A.** Artificial neural network for mobile robot topological localization. *Robot. Auton. Syst.*, 1995, **16**, 73–80.
11 **Yang, S. X.** and **Meng, M.** An efficient neural network approach to dynamic robot motion planning. *Neural Netw.*, 2000, **13**, 143–148.
12 **Braganza, D., Dawson, D. M., Walker, I. D.,** and **Nath, N.** A neural network controller for continuum robots. *IEEE Trans. Robot.*, 2007, **23**, 1270–1277.
13 **Pallottino, L., Scordio, V. G., Bicchi, A.,** and **Frazzoli, E.** Decentralized cooperative policy for conflict resolution in multivehicle systems. *IEEE Trans. Robot.*, 2007, **23**(6), 1170–1183.
14 **Ray, A. K., Behera, L.,** and **Jamshidi, Mo.** Sonar-based rover navigation for single or multiple platforms: forward

safe path and target switching approach. *IEEE Syst. J.*, 2008, **2**(2), 258–272.

15 Zhang, Q., Shippen, J., and **Jones, B.** Robust backstepping and neural network control of a low quality nonholonomic mobile robot. *Int. J. Mach. Tools Manuf.*, 1999, **39**, 1117–1134.

16 Das, T. and **Kar, I. N.** Design and implementation of an adaptive fuzzy logic-based controller for wheeled mobile robots. *IEEE Trans. Control Syst. Technol.*, 2006, **14**(3), 501–510.

17 Haykin, S. *Neural networks a comprehensive foundation*, 2nd edition, 1999 (Prentice-Hall, Englewood Cliffs, NJ).

18 Parhi, D. R. *Navigation of multiple mobile robots in an unknown environment.* Doctoral Thesis, Cardiff School of Engineering, University of Wales, UK, 2000.