International Journal of Electrical, Electronic and Communication Sciences
ISSN: 2517-9438
Vol:1, No:6, 2007

Mobile Robot Navigation Using Local Model Networks

Hamdi. A. Awad and Mohamed A. Al-Zorkany

Abstract — Developing techniques for mobile robot navigation constitutes one of the major trends in the current research on mobile robotics. This paper develops a local model network (LMN) for mobile robot navigation. The LMN represents the mobile robot by a set of locally valid submodels that are Multi-Layer Perceptrons (MLPs). Training these submodels employs Back Propagation (BP) algorithm. The paper proposes the fuzzy C-means (FCM) in this scheme to divide the input space to sub regions, and then a submodel (MLP) is identified to represent a particular region. The submodels then are combined in a unified structure. In run time phase, Radial Basis Functions (RBFs) are employed as windows for the activated submodels. This proposed structure overcomes the problem of changing operating regions of mobile robots. Read data are used in all experiments. Results for mobile robot navigation using the proposed LMN reflect the soundness of the proposed scheme.

Keywords**:** Mobile Robot Navigation, Neural Networks, and Local Model Networks.

1. Introduction

NAVIGATION of Mobile Robots is a broad topic, covering a large spectrum of different technologies and applications. Levitt, and Lawton [1] summarized the general problem of mobile robot navigation by three questions: "Where am I?," "Where am I going?," and "How should I get there?." This paper surveys the state-of-the-art in sensors, systems, methods, and technologies that aim at answering the first question, that is: robot positioning in its environment. Researchers in the area of mobile robot navigation have surveyed two main navigation approaches [2]. One is functional, or horizontal decomposition. The other is behavioural or vertical decomposition. The former approach is sequential and involves modeling and planning. The latter approach is parallel and requires exploration and map building. Both approaches use many distinct sensory inputs and computational processes. Decisions such as turn left, turn right, run or stop are made on the basis of those inputs [3].

Mobile robots have a large number of applications in industry, hazardous environments, and surveillance. Using the environmental information perceived at each instant as well as data from previous instants, a strategy should be pursued to enable the robot to reach its target position without hitting obstacles. Researchers have used many techniques for obstacle avoidance [2] and [4]. The basic task in such application is the perception of the environment through one or more sensors. Some authors have proposed that one type of sensor devices such as sonar, laser, vision and infrared be adopted [5] and [6], whereas others have recommended heterogeneous systems using different types [7]. Based on sensors information a complete conversion algorithm for unstructured environment was proposed [8].

Within the last decade, there has been an interest in being able to co-ordinate multiple mobile robots. This interest has stemmed both from practical considerations - multiple robots are able to handle tasks that individual machines cannot, for instance carrying large, bulky and heavy loads - and from a desire to create artificial systems that mimic nature in particular by exhibiting some of the primary behaviors observed in human and other animal societies.

A neural network is a parallel distributed processor comprising several simple computational units known as neurons [9], [10], [11], and [12]. Neural networks have generated considerable interest as an alternative nonlinear modeling tool [13]. They have some limitations. The limited structure of a neural network is the major limitation in its function approximation property. This means that new information may erase the previously learnt one [14].

Since most industrial processes operate under feedback control within a small region around a given operating point, neural networks with global support functions such as the Multilayer Perceptron (MLP) are sufficient for modeling and control them, however, if the operating point is changed, the function approximation of MLP is degraded [15]. Replacing the global support function (sigmoid function) with local support function (Gaussian function), results networks with local support function as radial basis function (RBF).

Recent research has suggested that fine partitioning of the input space, in which neural basis functions active, is not necessary. It is sufficient that the portioning procedure should simply split the input space into the expected operating regions of the plant [16]. The paper employs a local model network (LMN) that adopts this philosophy by forming a global system model from a set of locally valid submodels for mobile robot navigation. The idea was borrowed from [17]. The paper proposes fuzzy c-means (FCM) [18] for portioning the input space and discriminating the data of each submodel of the LMN. The outputs of each submodel are passed through a local processing function that effectively acts to generate a window of validity for the model in question. The resulted localized outputs are then combined as a weighted sum at the model output node. Each submodel is a MLP. Back propagation algorithm is employed to adapt the weights of each submodel. The proposed scheme was compared with the feedforward neural network scheme and RBF scheme.

This paper is organized as follows: Section II describes the mobile robot model. Section III briefly explains a set of neural networks used. The proposed LMN is detailed in section IV. Results using real data are depicted in section V.

II. Mobile Robot Model

A mobile robot is an autonomous vehicle that navigates in an environment to perform certain tasks. The mobile robot used in this research has three wheels. The two front wheels (left and right) are powered by separate stepper motors, and the rear wheel is free. The robot has an array of ultrasonic sensors for measuring the distances of obstacles around it and an infrared sensor for detecting the bearing of the target. The distances between the robot and obstacles act as repulsive forces, and the bearing of the target acts as an attractive force. The model of the employed mobile robot is obtained practically by recording a set of input-output data pairs using its sensors [2]. Multiple mobile robots are uses a set of robots to achieve complex tasks as mentioned above. Each mobile robot acts as an obstacle for other mobile robots. Each data set comprises of four inputs left distance, front distance, right distance, and target angle, and one output that is steering angle of mobile robot.

Analysis of mobile robot co-ordinates and directions used for adaptive navigation briefly reviewed [2]. The co-ordinates of robot '*n*' at time *t* are $x_n(t)$ and $y_n(t)$, and $v_n(t)$ is its velocity. The start and goal points of the robots are (x_{n0}, y_{n0}) and ($x0_n$, $y0_n$) respectively. Velocity directional angles of the robot are $\theta_n(t)$, where $(0 \le \theta_n(t) \le 2\pi)$ and $\theta_n(t)$ is measured from the X axis. The initial value is $\theta_n(0)$. The equation of motion of the n^{th} robot is given as:

$$\left.\begin{aligned} \dot{x}_n(t) &= v_n(t)\, Cos(\theta_n(t)) \\ \dot{y}_n(t) &= v_n(t)\, Sin(\theta_n(t)) \end{aligned}\right\} \quad (1)$$

For obstacle avoidance and navigation, the modulus of angular velocity $\dot{\theta}_n(t)$ is taken to be less than or equal to the maximum angular velocity:

$$\left|\dot{\theta}_n(t)\right| \le \frac{v_{n\max}}{r_{n\min}} \quad (2)$$

It is assumed that robots turn left or right with a minimum rotation radius $r_{n\min}$ (0.4m) because of their physical dimensions. $v_{n\max}$ (0.35 m/Sec.) is the maximum velocity of robots.

If there were no obstacles, the robots would instantly turn towards their goals at the start. This would be the optimal path for each robot but is impossible due to the robot dynamics.

Therefore, the following navigation law is proposed:

$$\dot{\theta}_n(t) = \eta_n\left[\theta_n^*(t) - \theta_n(t)\right] \quad (3)$$

where η_n is a positive constant.

As the robot has to turn a maximum of π radians at a time, η_n is taken as $\left[v_{n\max} / \pi r_{n\min} \right]$.

For better understanding, the navigation of robots without obstacles is now considered. Let $\theta_{nt}^*(t)$ be the desired angle for navigation of the robot to the goal i.e. $\theta_n^*(t) = \theta_{nt}^*(t)$ where,

$$\theta_{nt}^*(t) = \begin{cases} \phi_n(t) + \pi & \phi_n(t) \le \theta_n(t) \\ \phi_n(t) - \pi & \phi_n(t) > \theta_n(t) \end{cases} \quad (4)$$

$$\phi_n(t) = \tan^{-1}\frac{y_n(t)}{x_n(t)}, \quad 0 \le \phi_n(t) < 2\pi \quad (5)$$

$\phi_n(t)$ is the position angle of the target.

If a robot detects a target in the beginning (in a no-obstacle scenario) then

$$\dot{\theta}_n(0) = \eta_n\left[\theta_n(0) - \theta_{nt}^*(0)\right] \quad (6)$$

However, since $\phi_n(t)$ changes with the coordinates of the robot, ($x_n(t)$, $y_n(t)$), $\theta_{nt}^*(t)$ also changes with time.

Three types of distance sensors are used for the adaptive navigation technique described in this paper [2]. They are: d_{nc}; Central –front, d_{nl}; Central-Left, d_{nr}; Central-Right, d_{nl} and d_{nr} are inclined to the center at an angle α (this equals 30^o in the experiments carried out) to the central sensor d_{nc}.

The following assumptions have been made.

The maximum measurable range of the distance sensor is $d_{n\max}$ (taken as $3r_{n\min}$). When the sensor does not detect an obstacle or the distance is greater than $d_{n\max}$, then the sensor output is mathematically taken as a negative value (–1).

There are eight possibilities for avoiding obstacles. For example, when the obstacles are detected in three directions and $d_{nl} \ge d_{nr}$, the robot should steer to the left. Letting ε be the angle, the robot should turn to the left by $(\pi/2 - \varepsilon)$. More detail can be found in [2].

III. Neural Networks in MobilE Robot Navigation

This section describes briefly, a set of neural networks such as MLP and RBF that are employed for controlling mobile robot, as shown in Fig.1.

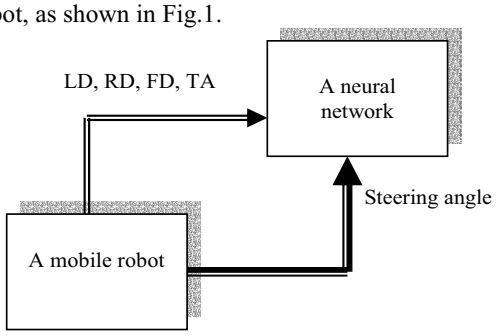

Fig. 1. Mobile robot navigation using neural network

A. Feedforward Neural Networks

Recently, feed forward neural networks e.g. MLP have been shown to obtain successful results in systems identification and control [13]. Such neural networks are static input / output mapping schemes that can approximate a continuous functions to an arbitrary degree of accuracy. The MLP neural network is composed of at least three layers, input, output and hidden layers. A neuron at a particular layer is connected to a set of neurons at the previous layer by a set of weights. The output y_i of a neuron is the weighted sum of the outputs coming from the neurons in the previous layer. That is :

$$y_i = 1/(1 + e^{-\lambda\, net_i}), \quad net_i = W_{ij}^T X \quad (7)$$

where $\lambda \in [0,1]$ is a weighting factor and W_{ij}^T is the weight that links the i^{th} neuron in a layer with the j^{th} input.

Backpropagation (BP) algorithm is employed to optimize the parameters of the network. When an input-output pair, *(u(k), y_d(k))*, is presented to the network at time *k*, the error function at the network output is defined as:

$$E_k = \frac{1}{2}\left[y_d(k) - y(k)\right]^2 \quad (8)$$

where $y_d(k)$ and $y(k)$ are the desired and the actual outputs respectively.

The general weight modification in the gradient descent method is:

$$\Delta w = -\eta \frac{\partial E_k}{\partial w} \quad (9)$$

where $\eta \in [0,1]$ is the learning rate.

B. Radial Basis Function Networks

Radial Basis Function network is a local network. That is that its universe of discourse is covered with a set of locally tuned radial basis functions. Research has been conducted using Radial Basis Function (RBF) networks in many areas such as pattern recognition, system identification and control and signal processing [14] and [17]. The main advantage of using this structure of network is that training will involve linear optimisation and can be performed on-line [19]. Structurally, it consists of three layers named the input, output and hidden layers. The RBF network simply approximates a continuous mapping, $f: R^n \rightarrow R^m$, by linearly combining a set of non-linear basis functions. This mapping can be described as:

$$y_{k(x)} = \sum_{i=1}^{N} W_k^i * \rho^i\left(\left\|x - c^i\right\|\right) \quad (10)$$

where $y_{k(x)}$ is the k^{th} output of the network, N is the number of hidden units, $x \in R^n$ is an input vector, $c^i \in R^n$ is the centre of the i^{th} hidden unit, $\rho^i{}_{(.)}$ is the i^{th} radial basis function and W_k^i is the weight from the i^{th} hidden unit to the k^{th} output.

Although there exist many choices of $\rho^i{}_{(.)}$, Moody and Darken [17] have reported that Gaussian type functions have the desirable feature of allowing the hidden units to be locally tuned. The Gaussian function is:

$$\rho^i(x) = \exp\left[-\left\|x - c^i\right\|^2 \Big/ \left(\sigma^i\right)^2\right] \quad (11)$$

where $\left\|x - c^i\right\|$ is a distance measure, usually taken to be the Euclidean norm. Each basis function, $\rho^i{}_{(.)}$, is centred at some point, c^i, in the input-output space. Because of the local nature of this type of function, it was employed in this work. The property is controlled by the variance, σ^i, which is a vital factor that speeds up learning. The RBF network has its origin in function approximation techniques and has a well-established theoretical basis [20]. Research has also been done to optimise RBF networks. Optimisation techniques were employed to adjust the parameters of the RBF networks, with linear least-squares optimisation applied to determine the height of an RBF, and the k-means clustering and nearest-neighbour algorithms used to locate the centre, c^i, and the variance, σ^i [21]. However, the latter algorithms yield poor centre and width (variance) selection and hence the overall performance of the controlled process will not be optimal. To achieve good results in this situation all RBF parameters have to be modified via a non-linear optimisation operation in the same manner as that encountered with MLPs. This means that the search direction is basically the negative of the gradient of a cost function at a particular weight. BP is a simple gradient descent algorithm with poor convergence properties resulting in long training times for MLP networks.

IV. Local Model Networks

The proposed LMN shown in Fig. 2.1. is a set of submodels depicted in Fig. 2.2. that represent a dynamic system be modeled at different operating points. Each submodel is a MLP network described in section III. The output of these local models are passing through a RBF that stands as a window for an activated model. The outputs of the fired submodels are weighted to give the total LMN output. These section describes the learning algorithm of MLP submodel and the clustering algorithm used to identify the data of each submodel.

First, two phases, the learning phase and the recall phase, are employed to build a particular LMN. The former consists of two learning segments, one structural and the other parameter-based. The former structured using the FCM that identifies a submodel of the LMN around its operating point. This is our first contribution in this paper. The submodels of the LMN are trained using the BP algorithm. A trained neural network basically represents a static knowledge base as mentioned above. Some important features have been added to the traditional neural network to yield evolving connectionist systems. These features are on-line learning, knowledge base adaptation and incremental learning. Learning schemes in neural networks can be classified generally into three categories, supervised, reinforcement and unsupervised. This paper employs the BP algorithm as supervised algorithm and FCM algorithm as unsupervised algorithm to construct the proposed LMN.

In the supervised scheme, a system should be directed by an external signal to achieve a desired performance. A common supervised learning method, the error BP scheme, is based on the steepest descent method. The major drawback of this method is its slow speed of learning and the local minimum problem that makes it not suitable for real-time application. Adding a momentum term to this scheme can sometimes stabilise and speed up the network convergence in training. A number of powerful second-order techniques have also been proposed to accelerate MLP as mentioned previously. The BP algorithm with these modifications has successfully been used in many networks, e.g. MLP. Mathematically, each weight, w_{ij}, is modified according to the following equation [11]:

$$\Delta w_{ij(k+1)} = -\zeta * \frac{\partial E_{(k+1)}}{\partial \Delta w_{ij(k+1)}} + \mu * \Delta w_{ij(k)} \quad (12)$$

where $E = \frac{1}{2}\left(y_d - \hat{y}\right)^2$, y_d and $\hat{y}$ are the desired and actual outputs of the network and ζ and μ are the learning rate and the momentum coefficient respectively.

Second, conventional clustering algorithms use crisp memberships for allocating samples to clusters. Where the input sample is assigned to one and only one cluster or category. However, in practice, a sample may be assigned to more than one class. One of the well-known clustering algorithms that allow fuzzy memberships is the fuzzy c-means (FCM) clustering algorithm. This paper employs this algorithm to segment the universe of discourse of mobile robots. Fuzzy c-means is an iterative clustering approach. It

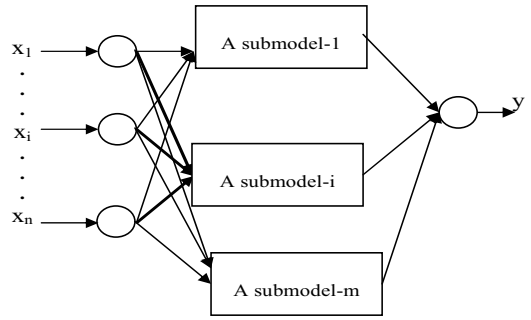

Fig.2.a. A typical structure of the proposed LMN

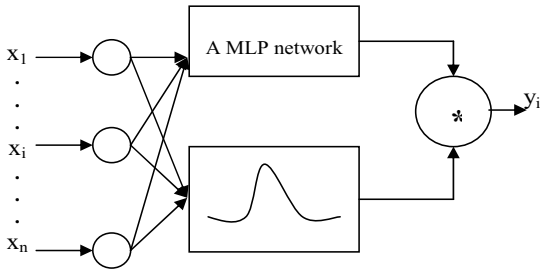

Fig.2.b. A submodel of the proposed LMN

resembles the maybe more well-known technique c-means but it uses fuzzy membership functions instead of hard values [17]. Fuzzy c-means partitions the data set $X = x_1,x_2,...,x_n$ into c fuzzy subsets u_i where the value $u_i(x_k)$ is the membership of x_k in class i. The values of $u_i(x_k)$ are arranged as a c x n matrix U. The method approximately minimises the sum of squared error function defined as:

$$J_m(U,V:X) = \sum_{i=1}^{c} \sum_{k=1}^{n} (u_{ik})^m \left\| x_k - v_i \right\| \qquad (13)$$

where $V= v_1,v_2,...,v_c$ is a set of cluster centers and $m>1$ is a weighting exponent affecting the fuzziness of u. The parameters (U,V) may minimise J_m only if u_{ik} and v_i are defined as:

$$u_{ik} = \left[\sum_{j=1}^{c} \left(\frac{\left\| x_k - v_i \right\|}{\left\| x_k - v_j \right\|} \right)^{2/(m-1)} \right]^{-1} \quad \text{for all i,k} \qquad (14)$$

$$v_i = \frac{\sum_{k=1}^{n} (u_{ik})^m x_k}{\sum_{k=1}^{n} (u_{ik})^m} \quad \text{for all i} \qquad (15)$$

The MLP submodel trained by BP algorithm can be identified a round its operating point using FCM that is employed for clustering the input domain of the mobile robot. A RBF performs a window for each submodel's output, given a higher degree ($\alpha \varepsilon [0,1]$) for the active submodel and a lower degree for the others. The LMN output can be described as:

$$y = f(\varphi,\phi) = \sum_{i=1}^{M} f_i(\varphi).\rho_i(\phi) \qquad (16)$$

where, $f_i(\varphi)$ represents a local submodel, M is the number of submodels, and $\rho_i(\phi)$ is a RBF defined in (11).

V. Testing

In this section, MLP, RBF and the proposed LMN have been implemented for controlling the mobile robot described in section II. The root mean square (RMS) error defined in (17) was computed for the trained networks using the test data set.

$$RMS = \sqrt{\frac{1}{T} \sum_{i=1}^{T} (y_d(k) - y(k))^2} \qquad (17)$$

where $y_d(k)$ and $y(k)$ are the desired and actual outputs respectively, and T is the number of samples during the run time. Design a particular mobile robot usually uses the expertise and common sense [22]. In this experiments, a 500 data set used in learning different types neural networks used in this paper [2], each data set comprises of five parameters, left distance (LD), front distance (FD), right distance (RD), target bearing (TA) and change in steering angle (SA). Fig. 3 shows a real robot used to generate the input output data depicted in Table I. More details for the hardware configuration can be found in [2].

In this experiments, one mobile robot, one target and four obstacles are employed for comparison result. For a set of robots, each mobile robot acts as an obstacle for the other robots as mentioned in section II. The experiments simulated at the same conditions mentioned above using the practical data sets on a PC operating under WINDOWS NT/95/98/2000/XP. The PC type is AMD Athlon 900MHz, cache memory 256k and RAM 64 k.

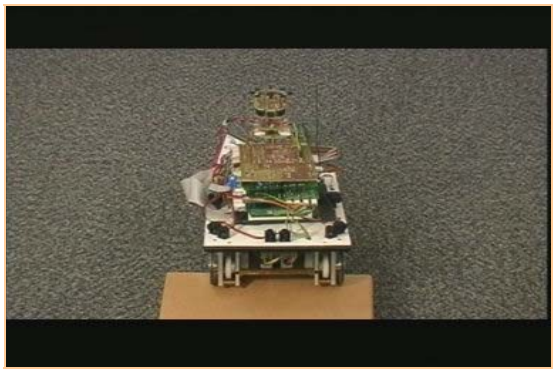

Fig. 3. A real mobile robot

A. MLP-based Mobile Robot Navigation

The MLP neural network used in this work has three layers, input, output and hidden layer. The input layer has four neurons, three for receiving the values of the distances from obstacles in front and to the left and to the right of the robot and one for the target bearing. If no target is detected, the input to the fourth neuron is set to 0. The output layer has a single neuron, which produces the steering angle to

TABLE I
REAL DATA SET FOR MOBILE ROBOT NAVIGATION [2]

LD	FD	RD	TA	SA
…	…	…	…	…
14	37	15	10	6
14	36	15	10	6
14	35	15	10	6
14	34	15	10	6
14	33	15	10	6
…	…	…	…	…

control the direction of movement of the mobile robot. The BP algorithm described in section III used for training the MLP network. Fig. 4 shows the mobile robot navigation in a bounded space using a set of obstacles. It successes to avoid the obstacles and reach the target in 1142.2 m sec. The RMS error defined in (17) was computed. That is 0.62.

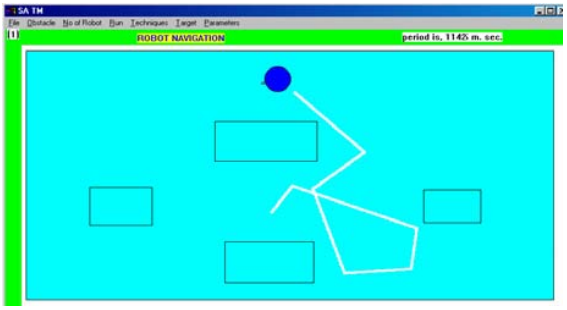

Fig. 4. The path traced by single mobile robot controlled using the MLP

This experiment was performed to multiple mobile robots navigation. For example Fig. 5 shows the four mobile robots navigation in a bounded space using a set of obstacles and one target. The time was taken by four robots navigation to avoid the obstacles, is 1492.1 m. Sec.

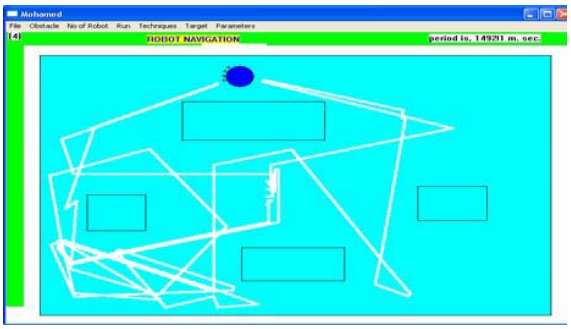

Fig. 5. The path traced by 4 mobile robots controlled using the MLP

B. RBF neural network-based mobile robot navigation

A RBF network with one hidden layer is used in this paper to control the mobile robot navigation. For comparison reasons, the network was tested at the same MLP structure and conditions. The RMS value defined in (17) was computed. That is 3.4. Increasing the hidden nodes of the RBF network, decreases the RMS value. For example, 222 neurons at the hidden layer of the RBF network gives a similar RMS value (0.6) obtained by MLP network. The mobile robot navigation using RBF network shown in Fig. 6.

It successes to reach the target in 53481.6 m. Sec. at 0.6 RMS value. For multiple mobile robots navigation using RBF network, the robots take long time to reach the target and sometimes robots can't reach the robot. Fig. 7 shows the four mobile robots navigation in a bounded space using a set of obstacles and one target. The time was taken by four robots navigation to avoid the obstacles, is 70251.5 m. Sec. This is because RBF network ia a linear combination of a large set of nonlinear functions and the Gaussian function of RBF network responds only to a small region of the input space where the Gaussian is centered.

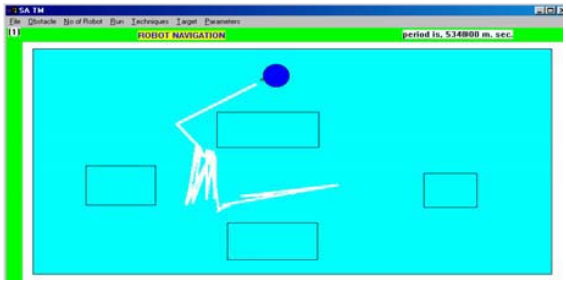

Fig. 6. Single mobile robot navigation using RBF

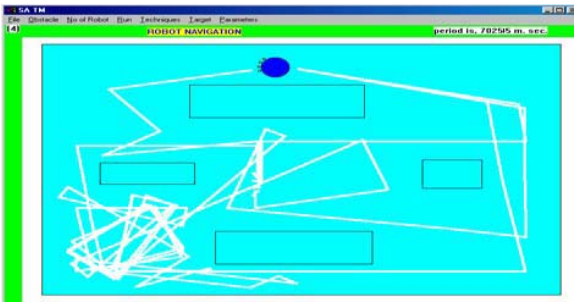

Fig. 7. The path traced by 4 mobile robots controlled using the RBF

C. LMN-based mobile robot navigation

The proposed LMN described above is applied on mobile robot navigation in this experiments. First, the data set of the input space was divided into two regions using FCM clustering technique described above. MLP described above are again employed as submodels. RBF defined in (11) used as a submodel window. Increasing the number of clusters (regions), increases the number of submodels, and decreases the RMS error. It also decreases the time needed for the robot to reach its target.

In the first experiment, Fig. 8 shows the mobile robot navigation in a bounded space using a set of obstacles. The number of clusters obtained using FCM was 11 clusters. The robot successes to avoid the obstacles and reach the target in 986.1 m. Sec. The RMS error defined in (17) was computed. That is 0.1. This experiment was performed to multiple mobile robots navigation. For example Fig. 9 shows the four mobile robots navigation in a bounded space using a set of obstacles and one target. The time was taken by four robots navigation to avoid the obstacles, is 1232.3 m. Sec.

Experimentally, we found that decreasing the number of training data set used for MLP learning, decreases the RMS error and the run time required to the robot to achieve its task. That leads to the proposed LMN trained by BP and employed FCM algorithm for clustering purposes. For comparison reasons, Table II shows the run time required for a set of mobile robot navigation to achieve their tasks.

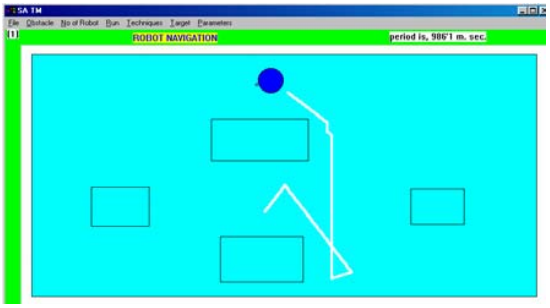

Fig. 8. Single mobile robot navigation using LMN

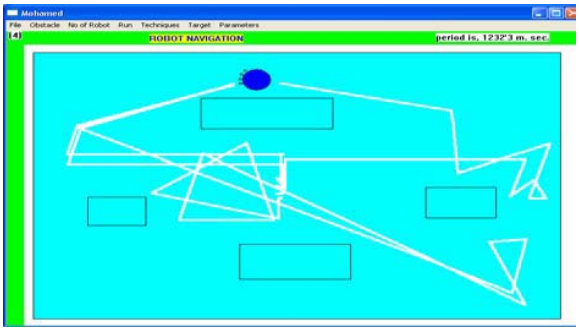

Fig.9. The path traced by 4 mobile robots controlled using the LMN

TABLE II
TIME TAKEN (m. Sec.) BY MULTIPLE MOBILE ROBOTS NAVIGIATION USING DIFFERENT SCHEMES

Number of mobile robots	Time taken using MLP	Time taken using RBF	Time taken using LMN
1	1142.20	53481.60	986.10
2	1361.99	60521.30	1121.30
4	1492.10	70251.50	1232.30
6	3085.84	80123.60	2523.19
8	4817.07	125645.70	2454.71

IV. Conclusions

The paper developed a local model network, that adapts the philosophy of forming a global system model from a set of locally valid submodels. These submodels are a set of MLP trained by BP algorithm. The paper also proposed the FCM to identify a submodel data set from the mobile robot domain. The proposed LMN was implemented for mobile robots navigation. Compared with MLP and RBF networks, the developed LMN scheme has three notable features, locality, generality and fast convergence. The former two features are stemmed from using submodels identified around the operating points of the robot be controlled. The latter is resulted from using RBF that identify the submodel in use quickly. These features make the proposed LMN promising scheme to mange with complex mobile robot navigation.

References

[1] T. S. Levitt and D. T. Lawton, "Qualitative navigation for mobile robots, "*Artificial Intelligence,* vol. 44, pp. 305-360, 1990.
[2] R. P. DAYAL, *Navigation of multiple mobile robots in an unknown environment*. Systems Engineering, University of Wales, Cardiff, 2000.
[3] O. Trullier, S. I. Wiener and A. Berthoz, "Biologicall based artificial navigation systems: review and prospects, " *Progress in Neurobiology,* vol. 5, pp. 483-544, 1997.
[4] N. Tomatis, I. Nourbakhsh, K. Arras and R. Siegwart, "A Hybrid approach for robust and precise mobile robot navigation with compact environment modeling, " *Proceedings of the IEEE*, 2001.
[5] L. Feng, J. Borenstein and H. R. Everett, "Where am I ? Sensors and methods for autonomous mobile robot positioning, "*Technical Report,* University of Michigan, MI, 1994.
[6] J. Borenstein and Y. Koren, "Noise reflection for ultrasonic sensors in mobile robot application, "*Proceedings of IEEE Conference on Robotics and Automation, Nice, France,* pp. 1727-1732, 1992.
[7] M. Buchberger, K. Jorg and E. V. Puttkamer, " Laserradar and sonar based world modeling and motion control for fast obstacle avoidance of the autonomous mobile robot MOBOT-IV, "*Proceedings of IEEE Conference on Robotics and Automation, Atlanta, Georgia,* pp. 534-540, 1993.
[8] E. Garcia and P. G. de Santos, "Mobile-robot navigation with complete coverage of unstructured environment, "*Robotics and Autonomous Systems,* vol. 46, pp. 195-204, 2004.
[9] I. Aleksander and H. Morton, *An Introduction to Neural Computing.* London: Chapman & Hall, 1990.
[10] K. J. Hunt, D. Sbarbaro, R. Zbikowski and P. J. Gawthrop, "Neural networks for control systems-A survey, "*Automatica*, vol. 28, pp. 1083-1112, 1992.
[11] S. Haykin, *Neural networks: A comprehensive foundation.* Macmillan college Publishing Company, Inc., USA, 1994.
[12] G. G. Lendaris and C. Paintz, "Training strategies for critic and action neural networks in dual heuristic programming methods,"*Proceedings of International Conference on Neural Networks 97, Huston*, pp. 712-717, 1997.
[13] I. Rivals, D. Canas, L. Personnaz and G. Dreyfus, "Modeling and control of mobile robots and intelligent vehicles by neural network, " *IEEE conference on intelligent Vehicles*, *Paris, France,* 1994.
[14] H. A. Awad, *Fuzzy neural networks for modeling and controlling dynamic systems.* Ph.D. Thesis, Cardiff School of Eng., Cardiff Unv., UK, 2001.
[15] A. Fink and O. Nelles, "Nonlinear internal model control Based on Local Linear Neural Networks, "*Proceedings of the 2001 IEEE, Man, and cybernetics Conference,* 2001.
[16] R. Murry-Smith, *A local model network Approach to nonlinear modeling.* PhD thesis, University of Strathclyde, Strathclyde, UK, 1994
[17] M. D. Brown, G. Lightbody and G. W. Irwin, "Nonlinear internal model control using local model networks, "*IEE Proc.-Control Theory Appl.*, Vol. 144, No. 6, November 1997.
[18] J. C. Bezdek, *Pattern recognition with fuzzy objective function-algorithms.* New York: Plenum Press, 1981.
[19] J. Moody and C. Darken, "Fast learning in networks of locally-tuned processing units, "*Neural Computation,* vol. 1, pp. 281-294, 1989.
[20] D. S. Broomhead and D. Lowe, "Multivariable functional interpolation and adaptive networks, "*Complex systems*, vol. 2, pp. 321-355, 1988.
[21] C. L. Chen, W. C. Chen and F. Y. Chang "Hybrid learning algorithm for Gaussian potential function networks, "*IEE Proc. D*, vol. 140, pp. 442-448, 1993.
[22] H. Huttenrauch, A. Green, M. Nornan, L. Oestreicher and K. S. Eklundh, " Involving users in the design of a mobile office robot, "*IEEE Transactions on Systems, Man, and Cybernetics-Part-C: Applications and Reviews*, vol. 34, pp. 113-124, 2004.

Hamdy Ali Ahmed Awad was born in Egypt, 1964. He received the B.Sc. degree in Industrial Electronics, and the M. Sc. degree in Adaptive Control Systems from Faculty of Electronic Engineering, Menouf, Menoufia University, Egypt in 1988 and 1994 respectively. He received the Ph.D. degree in Artificial Intelligent Systems in 2001 from School of Engineering, Cardiff University, Cardiff-Wales, England. Since July 2001.

He has been with the Faculty of Electronic Engineering, Menoufia University, where he is currently a lecturer of Control Engineering in Industrial Electronics Engineering and Control Dept.

Dr. Awad interests in intelligent control systems, machine learning, and their applications in control systems, fault diagnosis and medical engineering.