Neural Control System of a Mobile Robot

Moufid Harb, Rami Abielmona, Emil Petriu, and Kamal Naji

Aabstract—**Mobile robots could play a significant role in places where it is impossible for the human to work. In such environments, neural networks, instead of traditional methods, are suitable solutions to locally navigate and recognize the environment's subspaces. In order to learn and perform two important functions "environmental recognition" and "local navigation", multi-layered neural networks are trained to process distance measurements received from a laser range-finder. This paper will focus on a computer based design and test of this neural system, that includes three neural controllers for local navigation, and two neural networks for environmental recognition, fed off-line by a simulated model of a laser range-finder. These neural networks are the major components of a control system that performs a global neural navigation of a mobile robot, which could be used to perform industrial missions within industrial environments. This control system can guide a mobile robot to track its predefined path to arrive to its final goal through a set of sub-goals, or autonomously plan its path to arrive to the desired final goal, and to avoid obstacles that are found along the way.**

I. INTRODUCTION

With the progress of technology, autonomous navigation and path planning of mobile robots (MRs) have become a significant staple of the robotics industry. However, we have recently seen their migration into other domains such as drug delivery, search and rescue, as well as highway delineation. These new application areas require the navigation and manoeuvring of robotic structures within harsh, and possibly hostile, environments. Learning and adaptation have become two important features of any automated system meant to operate in environments that would present clear dangers for humans, such as places of high temperatures, deep space surroundings, and contaminated areas.

Conventional methods for controlling MRs show slow processing of data resulting from a sensory unit mounted on a MR, because of the serial processing of that data [1]. These methods need great efforts in programming to deal with nonlinear functions, as well as the difficulty of adapting to dynamic environments. Additionally, traditional approaches for robotic navigation are based on a detailed, accurate metric description of the environment [2], such as potential fields [3] and graph search methods [4]. Where it is difficult to perform in real-time applications [5], the prompt and high-speed processes for the input data are very important, especially if the kinematic constraints of the robot are taken into account [2]. Moreover, since the conventional methods represent the path in terms of artificial coordinate systems, they are highly vulnerable to spatial inaccuracies due to the sensory devices and movement actuators. Henceforth, neural networks are replacing traditional methods in the design of possible solutions to perform global navigation and path planning of a MR.

II. MODELING APPROACH

In this research, the modeling of a MR is executed using a personal computer. All processes of mobile movement control, measurements, mapping of working places, preparation and processing of databases were required for neural network (NN) training, while the comparison of results was performed by simulation using a high level programming language. The Matlab tool was used to train the NNs.

A. The Robot Model

As shown in Fig. 1, a differential drive vehicle was selected, where steering is affected by the difference in velocity of the tracked wheels. These wheels can be driven by two low-level DC motors located on either side of the center of the robot [6].

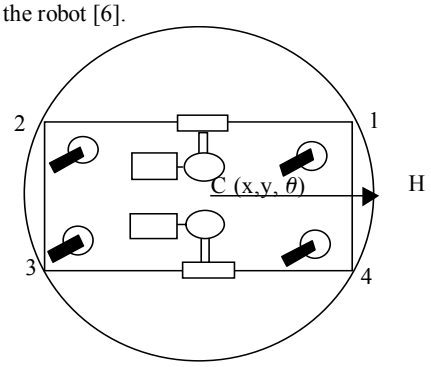

Fig. 1. MR model of differential drive vehicle

This type of MR has the ability to turn in place. The maximum dimension of the rectangular robot is 1 meter, which equals the diameter of the circle that surrounds the robot. Forward orientation angle θ and Cartesian coordinates X_C and Y_C determine the vector that represents the location of the MR. For distance measurements and direction of obstacles, a laser range finder [7] for the sensory system of a MR was suggested. To simulate the measurements of the proposed laser range finder, an algorithm [6] was used where

978-1-4244-1821-3/08/$25.00©2008 IEEE

the measured distances are relative to the MR dimensions (i. e. to the diameter of the circle that surrounds the robot). The correct modeling of the MR was proven by simulation tests that were conducted using the Simulink tool. The tests proved that the model of the MR is able to respond to the required changes in terms of both speed and direction [6].

III. ENVIRONMENTAL RECOGNITION

Fig. 2 shows the distribution of the selected measurements on the robot body. Nineteen directions regularly distributed with a separation angle of 15° surround the MR, aimed to cover most of the environment around the robot.

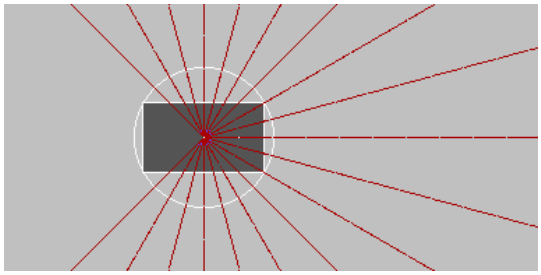

Fig. 2. Laser range-finder distance measurements for environmental recognition

In order to plan a mission path that a MR should follow, the construction map of the MR working place was considered to be available. The path tracking algorithm was then run while recognizing indoor subspaces whether the drive system is on-line or off-line. This path was classified according to a symbolic description of indoor subspaces, which was divided into a number of places such as corridors and cross- roads, etc.

A. Environmental Recognition NNs

A multi-layer NN model was chosen for this research [8]. It consists of three layers: the input, hidden and output layers, with a log-sigmoid function to activate the neurons. Two **environmental recognition NNs (ERNNs)** were designed: ERNN1 and ERNN2. Fig. 3 shows the input layer of ERNN1 that consists of 19 neurons, which associate with the output signals of the laser range finder of the MR's measuring system. The hidden layer consists of 19 neurons; this was found, through experimentation, to be the most suitable number for that layer.

Note that the number of neurons in the output layer represents the number of subspaces that are to be classified. These subspaces can be considered to be seven subspaces [2], or eleven subspaces [9].

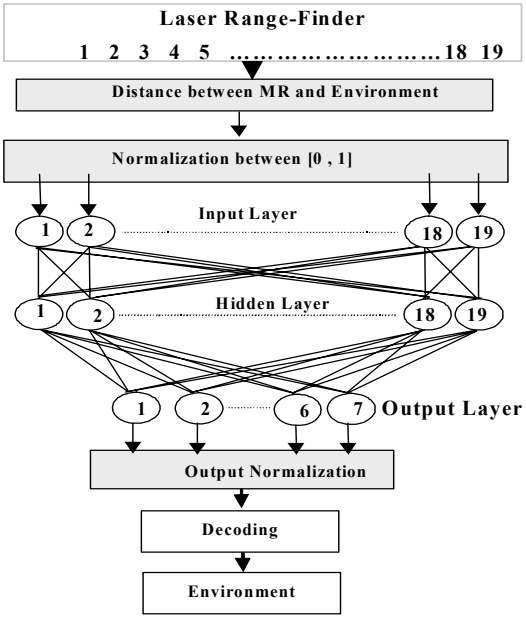

Fig. 3. NN1 structure diagram

Fig. 4 illustrates seven standard subspaces that are recognized by ERNN1, which are: corridor (CO), cross roads (CR), frontal T shaped cross-road (TF), left junction T shaped cross-road (TL), right junction T shaped cross-road (TR), left corner (LC), and right corner (RC).

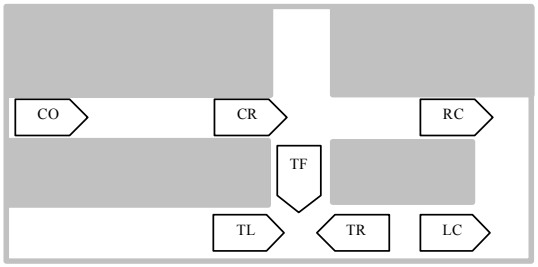

Fig. 4. ERNN1 classification subspaces

ERNN2 was designed and trained for environmental recognition to cover the other five subspaces, which has the same number of layers as ERNN1 but differs in the number of the neurons in the output layer (five). These represent five subspaces that could face the MR during its movement, and are defined as follows: dead end (DE), exit (EX), entrance (En), angle wall (AW), and room (RM). Fig. 5 shows these subspaces. In this paper, we will concentrate on the training and testing of ERNN1. The reader is referred to [6] for a more detailed explanation of ERNN2.

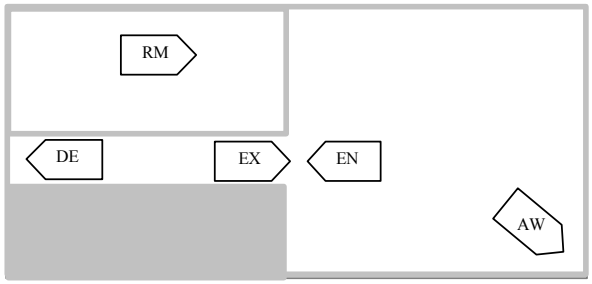

Fig. 5. ERNN2 classification subspaces

B. Data Base for Network Training

To train the NN, one must provide a suitable quantity of input vectors with the desired output vectors. This allows the NN to generalize previously unseen cases, as well as make correct approximations.

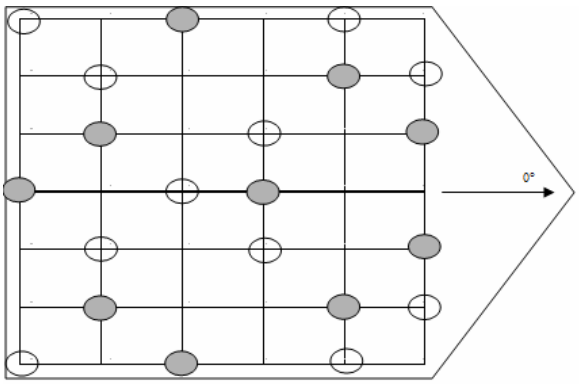

Fig. 6. Training and testing positions of NNs

Training areas were divided into a regular locations array. Eleven locations were selected for training in every subspace of the seven subspaces shown in Fig. 4. The orientation of the robot was varied by 15° within an angular sector of 90° (i. e. ± 45° from the front direction of the subspace as shown in Fig. 6).

C. ERNN1 Learning

To train the designed ERNN1, the back propagation learning algorithm was used because it is suitable for multi-layered NN learning. To avoid the problem of local minima, the technique of back propagation with momentum was used. This is due to its better and faster performance when compared to the standard back propagation algorithm [10]. A random function was selected to generate initial small weights suitable for ERNN1.

A learning rate of 0.001 was introduced to avoid losing the main learning direction when introducing unusual training vectors to the NN [11]. After 31000 training epochs, ERNN1 converged with a minimum possible error, where the sum squared errors (SSE) was 15.17.

D. ERNN1 Performance Test

To check the capability of ERNN1 for generalization, the robot was placed at differing locations from those used for training, as shown in Fig. 6. Ten locations were selected in every subspace. The robot was directed into three frontal directions in every location, -45° right, 0° front, and 45° left, assuming 0° is the front direction of the subspace as shown in Fig. 1.

TABLE I.
CRITICAL CASES OF ERNN1

Actual Place	ERNN1's actual output							ERNN1's desired output						
	7	6	5	4	3	2	1	7	6	5	4	3	2	1
CO	0	0	0	0	0	0	0.25	0	0	0	0	0	0	1
CR	0	1	0	0	0	0.01	0	0	0	0	0	0	1	0
CR	0	0.08	0.01	0	0	0.29	0	0	0	0	0	0	1	0
CR	0	0	0.13	0	0	0.43	0	0	0	0	0	0	1	0
CR	0	1	0	0	0	0	0	0	0	0	0	0	1	0
CR	0	0.99	0	0	0	0.05	0	0	0	0	0	0	1	0
RC	0	0	0	0.15	0.28	0	0	0	0	0	0	1	0	0
TR	0	0	0.58	0	0	0.14	0.01	0	0	1	0	0	0	0
TR	0	0.75	0	0	0	0	0	0	0	1	0	0	0	0
TR	0	0	0.05	0	0	0.01	0	0	0	1	0	0	0	0
TL	0	0.09	0	0	0	0	0	0	1	0	0	0	0	0
TL	0	0.02	0	0	0	0	0	0	1	0	0	0	0	0

The performance of ERNN1 was tested, with table I showing the critical locations for ERNN1. This table provided insightful information concerning the locations that ERNN1 shall be trained at. The best result was achieved for the TF. The poorest result was for the CR, due to the MR's proximity to the wall where the scanning sensory sector of the robot was quite small. Therefore, additional training measurement vectors of the critical locations were introduced to ERNN1 to allow for a more complete learning cycle.

IV. LOCAL NAVIGATION

For the local navigation of a MR, three neural controllers were designed and trained. These controllers are termed: "Keep Right", "Keep Left", and "Pass a Cross-Road", where:

1) The "Keep Right" controller is to keep the robot at the right side of the surroundings,
2) The "Keep Left" controller is to keep the robot at the left side of the surroundings, and
3) The"Pass a Cross-Road" controller is to pass a cross road.

Furthermore, these neural controllers avoided obstacles, which were found in a MR's way. They also received commands from a decision maker that played the role of global navigator, which will be discussed in section V. For

the purpose of local navigation and to achieve higher resolution in distance measurement, the measuring range of the laser range-finder was limited to 3.5 meters. This helped in distinguishing the nearby obstacles, when the measurement vector was normalized to a value between 0 and 1.

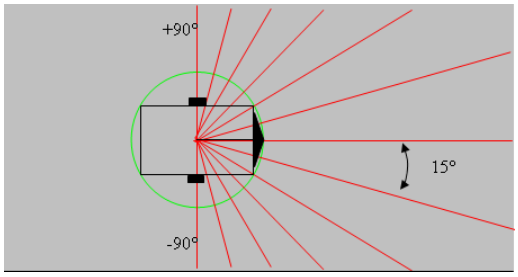

Fig. 7. Laser range-finder distance measurements for local navigation

As shown in Fig. 7, thirteen measurements were taken within a measuring sector of –90° to 90° to the frontal direction of the robot, with intervals of 15°. These measurements were fed to the local navigation neural controller (LNNC) that has the structure shown in Fig. 8.

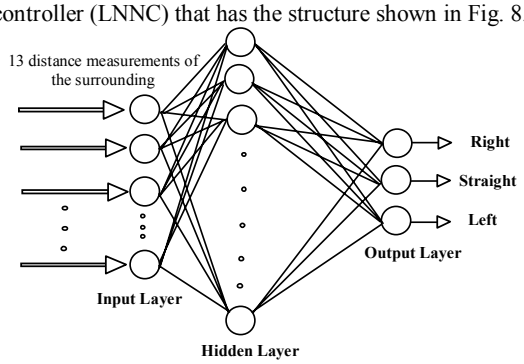

Fig. 8. General Structure of the LNNCs

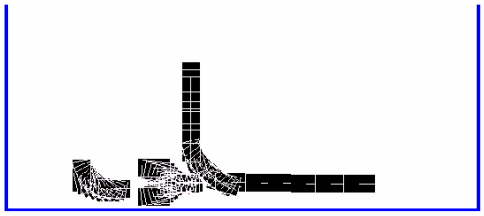

Fig. 9. Follow wall training locations and directions

A. Design of the "Keep Right" Neural Controller

To train the "Keep Right" LNNC, a database was collected by locating the MR on 111 positions as shown in Fig. 9 and Fig. 10. The required steering signal was considered to move the robot a step, while trying to keep it at a safe position from the obstacles. The database contained the training vectors, where every vector had two parts. The first part represented the sensory input of the LNNC shown in Fig. 8. These are the thirteen distance measurements of the laser range-finder. The second part was the desired output which represented the steering signal in binary code, for example (1 0 0: is Turn Left), (0 0 1: is Turn Right), and (0 1 0: is Go Straight).

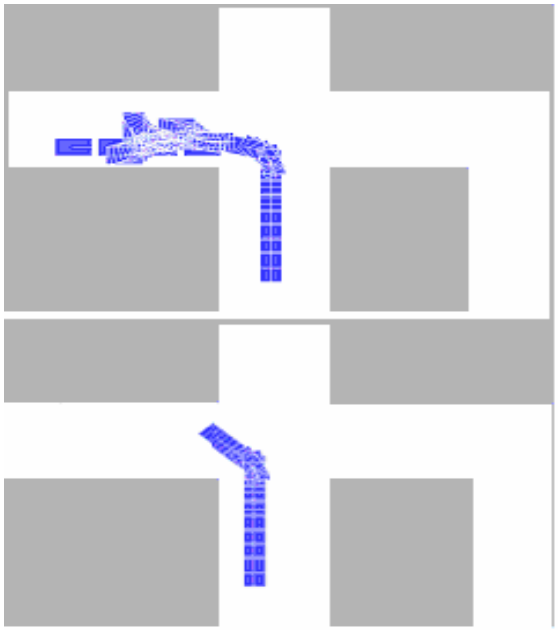

Fig. 10. Turn right at cross-road training locations

Furthermore, to train the "Keep Right" LNNC, the Matlab Neural Networks Tool Box was used to run the back propagation algorithm with momentum and adaptive learning rate [8]. Fig. 11 shows the strategy of the training technique.

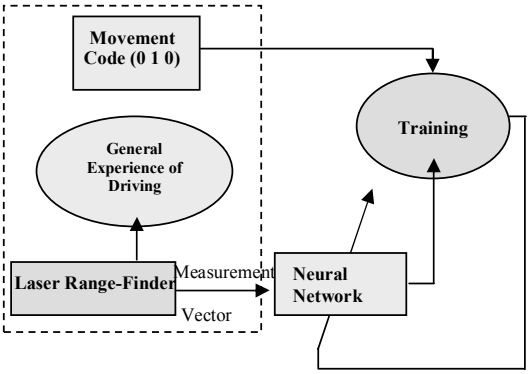

Fig. 11. Training of the LNNCs

After 92000 training epochs, the weights converged when the total SSE was equal to 2.000011. The test was done for the "Keep Right" LNNC in unknown environments, and it proved to exhibit a good performance as shown in Fig. 12.

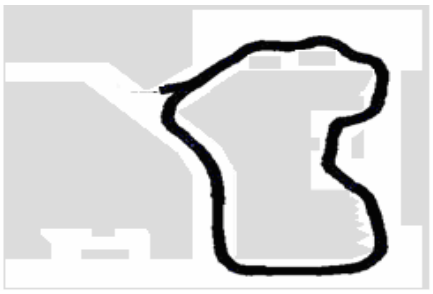

Fig. 12. "Keep Right" LNNC test in unseen environment

B. Design of the "Keep Left" LNNC

To design the "Keep Left" LNNC, it was easier to use the "Keep Right" LNNC to generate the appropriate steering signals by spatially mirroring the sensory inputs of the "Keep Right" neural controller, and changing the sign of the steering signal [2]. Fig. 13 shows the algorithm used to convert the "Keep Left" LNNC.

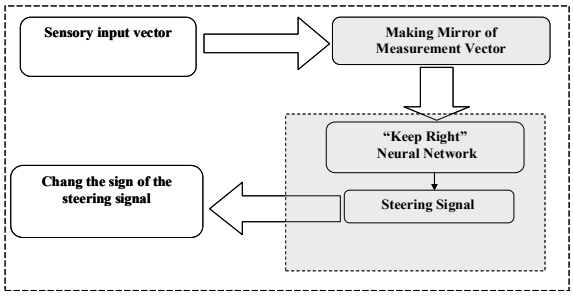

Fig. 13. Algorithm to convert to "Keep Left" LNNC

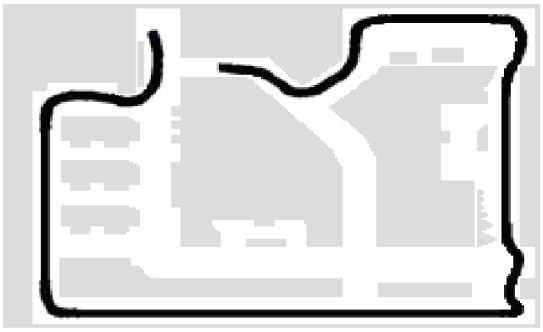

Fig. 14. "Keep Left" LNNC test at in unseen environment

As shown in Fig. 14, the test was conducted for the "Keep Left" LNNC in different environments and it proved to exhibit a satisfactory performance.

C. Design of the "Pass a Cross-Road" LNNC

The design structure of this controller is similar to the design of the "Keep Right" LNNC. The main difference is the training database introduced to the neural network to learn its task. Similar to the above-mentioned techniques, database training vectors were collected by locating the MR at 47 positions as shown in Fig. 15.

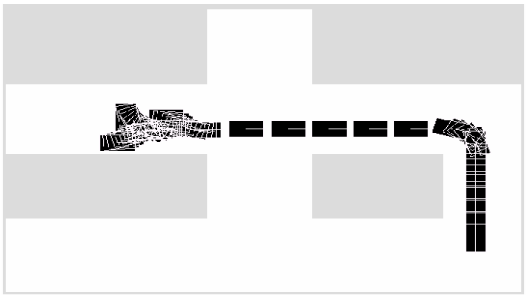

Fig. 15. Pass a cross-road and then keep right training locations and directions

The training vectors were enough to train the LNNC for passing a cross. The network weights converged after 43564 epochs. The SSE for the training vectors was equal to 1.10^{-6}, which was the targeted error. Fig. 16 shows the test of the "Pass a Cross-Road" neural controller. It can be noticed that the MR can pass the cross-road when it recognizes the cross.

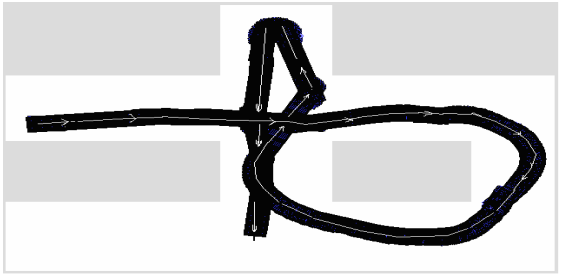

Fig. 16. "Pass a Cross-Road" test

V. GLOBAL NAVIGATION

Neural global navigation provided control of a MR's direction, planning of its path to the main goal, tracking a predefined path as sub-goals, and avoidance of expected and unexpected obstacles. Fig. 17 shows the block diagram of our neural control system designed and developed to achieve movement direction and control of a MR.

A. Decision Maker

The main duty of the decision-maker, shown in Fig. 17, is to achieve the global navigation that will move a MR from its current position to the main goal, track sub-goals found in

a predefined path, and finally get to the main goal. For local navigation missions, the decision-maker shall select between the aforementioned three LNNCs: "Keep Right", "Keep Left" and "Pass a Cross-Road". This selection is performed by two different methods: autonomously, or by following the symbolic predefined path description. Considering both methods, our first and second ERNNs were used. An additional three tasks: "go straight" for short distances, "turn angle", and "go to goal", were also used when a line of sight becomes clear between the MR and the main goal.

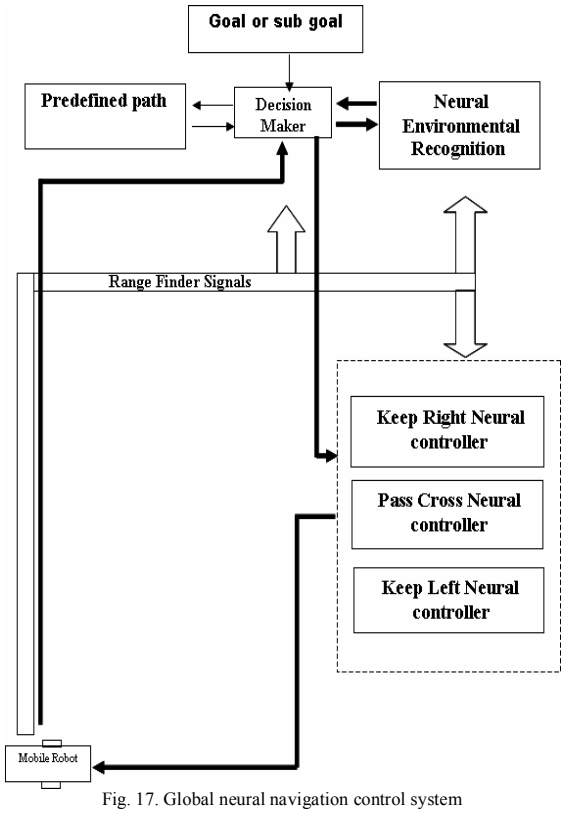

Fig. 17. Global neural navigation control system

B. *Predefined Path Planning*

For the path planning approach, a combination of two methods was used. The first method determined sub-goals, and the second method autonomously found the robot path. Considering the first method, we sequenced the path using twelve subspaces (i.e. nodes and connections), which was identified using seven subspaces in [12]. A sequence of subspaces can be specified by an expert after providing the industrial working environment map, and knowing the direction from where the MR will view the subspaces. These subspaces can then be recognized by the designed ERNNs, where a new algorithm with a different technique from Biewald [12] was used. Fig. 18 shows the environment and predefined path used in [12], but we tabulated the mission in a different way. table II shows the mission description where we used the classification of twelve subspaces that can be recognized by our ERNNs. The author in [12] considered the existence of a CO between every two subspaces. However, for the necessity of contrast and path tracking, we defined additional CO subspaces only in between every two adjacent similar nodes, such as nodes 24, 25, 26, and 27 shown in Fig. 18. Also, in [12], subspace 17, shown in Fig. 18, was considered as a CR, however it actually was an EX to a hallway. Furthermore, subspace 18 was defined as T shaped cross road in [12] , whereas we defined it as an EN.

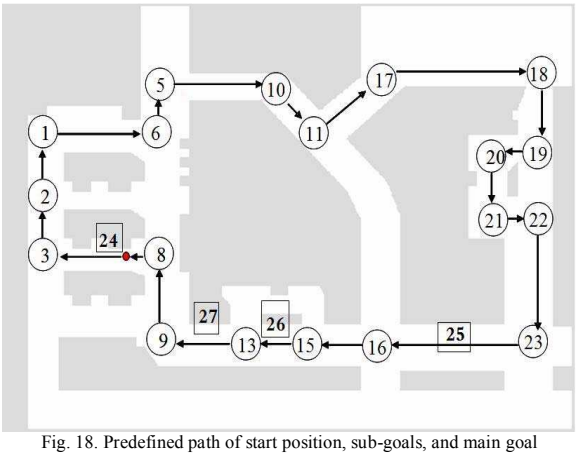

Fig. 18. Predefined path of start position, sub-goals, and main goal

TABLE II.
PATH DESCRIPTION

LNNC	Selected ERNN	Subspace No. in Fig. 18	Subspace	Path
R	1	24	CO	1
R	1	3	TF	2
L	1	2	TR	3
R	1	1	RC	4
L	1	6	TF	5
R	1	5	TR	6
R	1	10	RC	7
L	1	11	TL	8
R	2	17	EX	9
R	2	18	EN	10
R	1	19	TR	11
R	1	20	LC	12
R	1	21	LC	13
R	1	22	TF	14
R	1	23	CR	15
R	1	25	CO	16
L	1	16	CR	17
S	1	15	TR	18
L	1	26	CO	19
L	1	13	TR	20
L	2	27	EX	21
L	1	9	TR	22
R	1	8	TL	23
L	2	24	CO	24

Fig. 19 shows the simulation performance test for the global neural navigation control system (GNNCS) at the factory environment based on the predefined path shown in Fig. 18 and described in table II. The GNNCS successfully achieved the mission and avoided the unexpected obstacles.

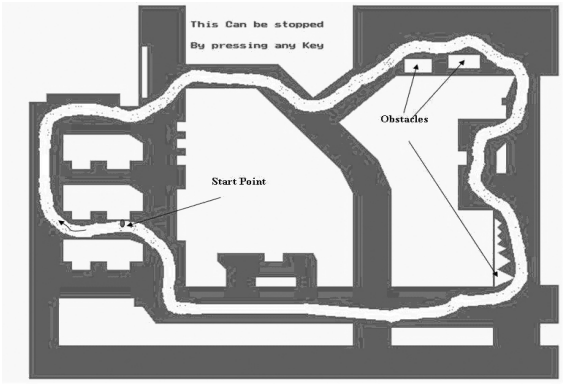

Fig. 19. The GNNCS performance at the factory environment based on the predefined path

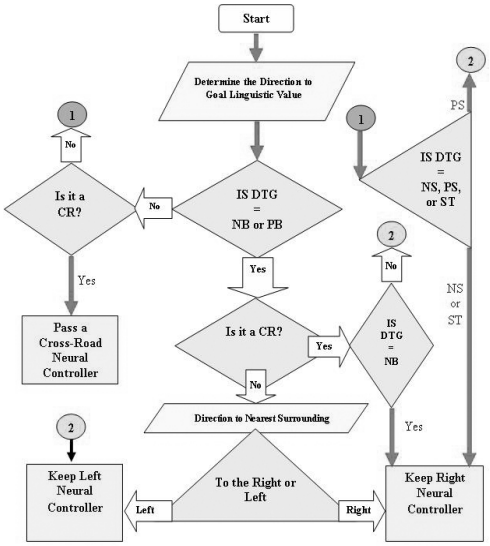

Fig. 20. Local navigation neural controller's selection process used by autonomous path planning algorithm

C. *Autonomous Path Planning*

Our second method was to autonomously get to the main goal from the present position of the MR. The decision-maker used our ERNN1 to check if the MR faced a cross-road and calculated its direction to the goal (DTG) angle with respect to the current frontal direction of the MR. These two factors helped to determine the neural controller for local navigation that must be used. Other factors were also considered such as the existence of obstacles between the MR and the main goal, as well as the distance between the MR and the main goal. table III shows the linguistic values of the angle between the MR's current frontal movement direction, and the direction from the current location of the MR and the main goal's location. In our simulation, we considered that the MR was equipped with a signal receiver that can detect coded signals transmitted from the goal location, or a MR's base (in the case of returning back to its base location after executing its mission) and decoded them to determine the direction to the main goal or to its base. Fig. 20 shows the process of selecting the local navigation neural controller that was needed to perform a mission assigned by the algorithm of autonomous path planning. This algorithm was tested using different cases as shown in Fig. 21 and Fig. 22, and as can be seen, it proved to possess the desired performance.

TABLE III.
LINGUISTIC VALUES OF THE DTG ANGLE

Linguistic values of									
Negative Big (NB)		Negative Small (NS)		Strait (ST)		Positive Small (PS)		Positive Big (PB)	
-180	-91	-90	-31	-30	30	31	90	91	180

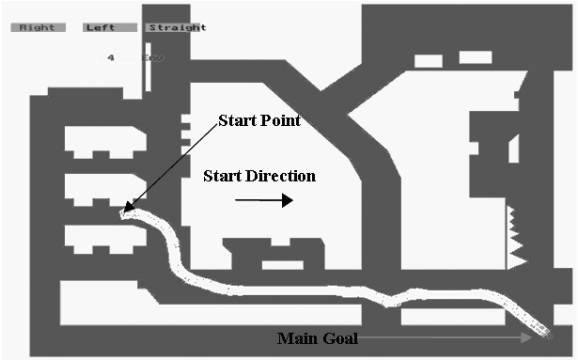

Fig. 21. GNNCS performance at the factory environment using autonomous path planning, this case considers the existence of a CR

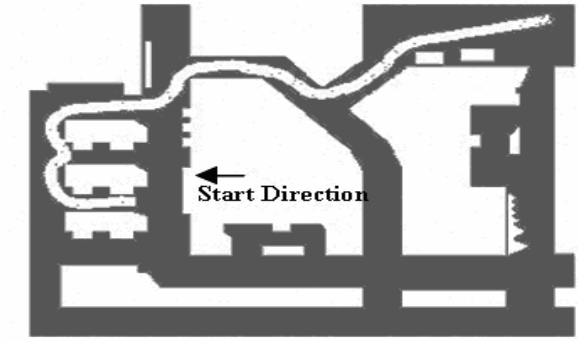

Fig. 22. GNNCS performance at the factory environment using autonomous path planning for different case

VI. CONCLUSION

Most mobile robotic systems are non-linear systems with kinematics that are very difficult to control by traditional methods, while, multi-layered neural networks have proved their competence to control mobile robotic systems, particularly when faced with real-time obstacle detection and avoidance.

The paper's main contributions are a novel training technique based on the method of creating a flat files database, and the number of utilized neural networks that were used in the control system of the MR. By comparison, the number of neural networks in [9] was equal to the number of subspaces to be classified, while in [12], to recognize only a total of 7 subspaces, one neural network was designed and trained by locating the mobile robot randomly in the training subspaces using the uniform distribution function. In this paper, the training was guided and the locations were previously determined, which differs from the technique used in [9] and [12]. The use of two neural networks, ERNN1 and ERNN2, allowed for a successful recognition of twelve subspaces.

The designed algorithm for controlling a MR is also a new approach, especially when a novel way to define a MR's path was developed and compared with what has been previously published. The simulation testing of both of our developed methods for path planning proved their efficiency and capability, in achieving the global navigation of the MR, controlling the direction of the MR to track its planned path, and reaching sub-goals and the main goal while avoiding obstacles.

This research also showed the significance of using neural techniques as an intelligent system to solve complex and critical control problems, where our five designed and trained neural networks were the base of the global navigation system. The two environmental recognition NNs were capable of recognizing the searched subspaces despite the fact that these subspaces were not in the standard shape that the NNs were trained on. Finally, the neural controllers for local navigation, "Keep Right", "Keep Left" and "Pass a Cross-Road" successfully performed in previously unseen environments.

Future improvements will focus on the integration of the GNNCS with a fuzzy logic system to create a hybrid fuzzy-neural control system in order to achieve the MR's speed control, as well as the GNNCS' utilization to resolve *simultaneous localization and mapping (SLAM)* problems that exist when autonomous robots traverse unstructured environments.

Future research includes the integration of the developed autonomous navigation scheme to the intelligent sensor agents (ISAs) presented in [13]. These agents are intended for the autonomous investigation of relevant parameters in natural living environments, industrial or laboratory hazardous environments, polluted environments, water treatment plants, nuclear stations, war zones, or remote difficult to reach environments such as mining and deep-sea exploration. As an individual ISA only offers local and limited information about the environment, multiple and diverse ISAs are needed to cover the large area and multi-parameter natural environments that are monitored in real-life situations. A robust local navigation and environmental recognition scheme is required for the resolution of the SLAM problem that exists within intelligent robotic agents traversing unstructured environments. The research presented in this paper provides a proven and deployable methodology for the control system of each of the ISAs.

REFERENCES

[1] Sklarenko, E. G., " Control system for robotic electric drive using Neural models", Conference of problems of automotive electrical drive, Yalta, Ukraine, pp. 385-387, 1998.

[2] Zalzala, A. M. S. and A. S. Morris, "Neural Networks for Robotic Control: Theory and applications", Ellis Horwood, 1996.

[3] Badcock, J.M. and Dun, J. A., "An autonomous robot navigation system-integrating environmental mapping, path planning, localization and motion control", Robotica Journal, volume 11, pp. 97-103, 1993.

[4] Antonio, J. , Madrigal, F., and González, J., "Multihierarchical Graph Search", IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 24 Issue 1, IEEE Computer Society, IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 24 , Issue 1, 2002.

[5] Habib, Maki K., "Real Time Mapping and Dynamic Navigation for Mobile Robots", International Journal of Advanced Robotic Systems, , ISSN 1729-8806, pp. 323-338 323, Vol. 4, No. 3, 2007.

[6] Harb, M., "Computer modelling of industrial mobile robot and control by incorporated neural networks and fuzzy logic", University of Damascus, Syria, PhD Thesis, 2000.

[7] Nof, S. Y., "Handbook of Industrial robotics, 2nd edition", John Willy & Sons, Inc., ISBN:0471177830, 1999.

[8] The Math Works, "Matlab User's Guide", Math Works, Inc.

[9] Al-Allan, S., "Environment recognition and reactive navigation of an autonomous mobile robot using neural networks", University of EVRY, France, Ph.D. Thesis, 1996.

[10] Kartalopoulos, S. V., "Understanding of Neural networks & Fuzzy logic, basic concepts and applications", IEEE Press, 1996.

[11] Fausett, L., "Fundamentals of Neural networks", Prentic Hall, 1994.

[12] R. Biewald, "A neural network controller for the navigation and obstacle avoidance of a mobile robot", In *Neural Network for Robotic Control*, A. M. Zalzala and A. S. Morris, Eds., Ellis Horwood, 1996.

[13] Abielmona, R., Petriu, E.M., and Whalen T., "Multi-Agent System Information Fusion for Environment Monitoring", Instrumentation and Measurement Technology Conference (IMTC), ISBN 0-7803-9359-7, pp. 1774-1779, April, 2007.