International Journal of Scientific & Engineering Research, Volume 4, Issue 5, May-2013
ISSN 2229-5518

Lowcost Mobile Robot using Neural Networks in Obstacle Detection

Nagarani R[*1], Nithyavathy N[*2] and Dr.Parameshwaran R[*3]
[*1]PG Scholar, Department of Mechatronics, Kongu Engineering College, Erode, Tamil Nadu- 638052
[*2]Assistant Professor, Department of Mechatronics, Kongu Engineering College, Erode, Tamil Nadu- 638052
[*3] Head of the Department, Mechatronics, Kongu Engineering College, Erode, Tamil Nadu- 638052
grsakthishree@gmail.com , nithyavathy@kongu.ac.in, paramesh_1@kongu.ac.in

Abstract— This paper describes the neural network based obstacle avoidance robot using low cost IR sensor arrays. IR sensors output are arranged in "0" and "1" combination. It constructs 256 input patterns. An intelligent controller like Neural Network is used to train the input pattern. Multilayer feed-forward back propagation algorithm is implemented. The Neural net is trained offline by using MATLAB coding. The trained pattern is interfaced with LabVIEW software. LabVIEW is a sophisticated tool for developing and running a real time application. This graphical user interface commands the robot to turn an angle where there is no obstacle. The sensor inputs and robot turning movements are controlled by using PIC 16F877A microcontroller.

Index Terms— Backpropagation, IR sensors, LabVIEW, MATLAB, mobile robot, microcontroller, neural network, obstacle detection.

1 INDRODUCTION

Mobile robot is most important research area in the robotic field. This helps to increase the intelligence of the robot to recognize the environment by the robot. But it has some real time difficulties, such as unknown obstacles, unguided path-way, etc. In order to overcome these difficulties the robot must have number of sensors to detect and correct these difficulties. This leads to increase the cost of the overall project. Anyhow it is developed in many areas like neural network, fuzzy logic, etc. with some costly sensors like ultrasonic. This project is also concentrated on the cost issue. This project used IR sensors for obstacle measurement; its cost is much cheaper than other sensors.

Neural Network (NN) is an information processing paradigm that is inspired by the way biological nervous systems by the way biological nervous systems, such as the brain, process information. The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems. An NN is configured for a specific application, such as pattern recognition or data classification, through a learning process.

Supervised learning process is implemented. Levenberg-Marquardt algorithm which trains a neural network 10 to 100 faster than the usual gradient descent Backpropagation method is established. Also we simplify the net which has only one hidden layer. The input pattern is trained using MATLAB coding. Tan sigmoid is the activation function for the hidden neurons and purelin for the output neuron. The network training function is Trainlm.

LabVIEW is a sophisticated tool for developing and running a real time application This Graphical programming language acquire the weight and bias values of the net trained offline by using MATLAB. The computational process is done. The final output which is an angle where the robot is turned is calculated.

In this paper, PIC16F877A microcontroller is implemented. By using PIC controller, sensors output are turned into "1" and "0" combination when obstacle is present or not. Also robot turning angle which are processed in PC is transferred via this controller. According to that the robot turns and finds its path.

Matt Knudson and Kalgan Turner [1] Oregon implement many algorithms like Rule-based and Neuro-evolutionary for robot navigation. It leads to the multiple algorithms to follow. Ultrasonic sensors are used which are cost effective. In this both simulation and experimental setups are done.

Antonio Sgorbissa and Renato Zaccaria [2] describe the roaming trails in which the obstacles are classified as static, moving and removable. Both on line and off line method are implemented. Also experimental setup has more hardware. Ehsan Hashemia, Maani Ghaffari Jadidi and Navid Ghaffari Jadidi [3] are implemented the combination of PI & Fuzzy controller for robot navigation. Four Omni-directional wheel alignments are used for obstacle avoidance. Neural and less wheel alignment is enough to implement.

Kai-Hui Chi and Min-Fan Ricky Lee [4] implement the neural network to train the pattern using MATLAB. This technique nearly suits the need but instead of sonar sensor, IR array for closed degrees are studied. Also we simplify the Neural Net by reducing the hidden layers. The results are shown virtually by interfacing LabVIEW environment. The experimental setup is installed with PIC controller and servo motor. In this way we implement the system in real time application. From the literature review we conclude it is simpler and more reliable.

2 METHODOLOGY

2.1 Robot Hardware Setup

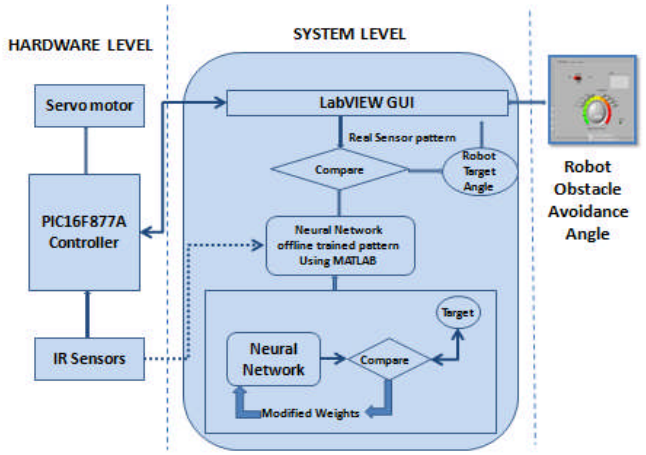
The hardware setup consists of 8 IR sensors with signal conditioning unit. They are arranged in front of the robot around 180° angle. The Conceptual design is shown as in fig 1.

Fig.1.Conceptual Design.

The sensor output will turn into "1", when the obstacle is sensed. Otherwise, it will turn into "0". Likewise 8 sensors output are arranged

as a pattern. For example if we consider there is no obstacle before the robot path. The pattern arranged will be [0 0 0 0 0 0 0 0]. In this way we are creating 2^8 = 256 patterns to help the robot to detect the obstacles.

The PIC16F877A microcontroller is implemented to acquire sensor



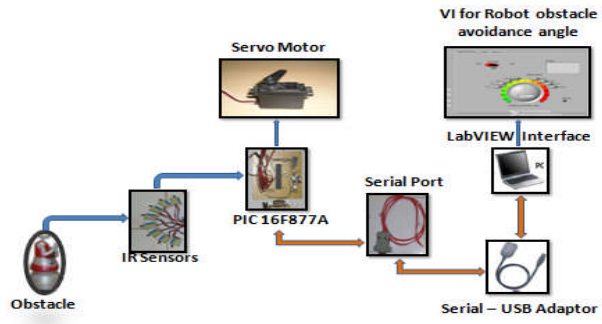
output pattern. These patterns are transferred to PC via serial port. The 256 combinations are already trained offline using neural network. Sensors output are processed and compared with the trained pattern. This neural computation is done by using LabVIEW. The angle detection where there is no obstacle is determined. This is shown virtually in PC using LabVIEW. The detected angle is sent to the servo module through PIC controller from PC. The knob placed at the top of the servo motor is turned according to the angle of detection. Likewise it is shown in real time. The Hardware setup is shown as in fig 2.

Fig 2 Experimental Setup

3 Neural Network Controller

3.1 Back Propagation Algorithm

The pattern is trained using neural network. Back propagation model is adopted. Back propagation neural network is derived from the delta rule. The input and the target vectors should be fed initially for training the network pattern. Training these networks is done by



changing the weights of the unit depend on the error occurring at that unit. The weight change rule is developed from the Perceptron learning rule. The output unit error is used to alter the weights on the output unit. The hidden layer errors can be calculated by back propagating the errors at the output unit and the hidden layer weights are altered with those errors. For each data set, the forward pass and backward pass is continued until the error become too low. Unlike other networks, in back propagation network the errors can be back propagated to the hidden layers, so that more accurate results can be obtained.This is shown in fig 3.

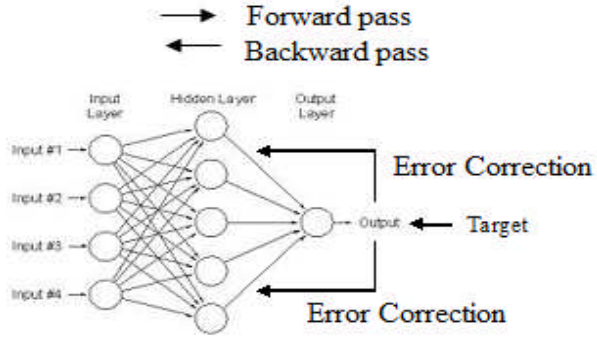


Fig 3 Back propagation neural network

The Backpropagation model is summarized as follows

The forward pass:

Input to the hidden unit,

$$Z_{inj} = V_{0j} + \sum_i x_i v_{ij} \qquad (1)$$

Output of the hidden layer,

$$Z_j = f(Z_{inj}) \qquad (2)$$

Input to the output unit,

$$Y_{ink} = W_{0k} + \sum_{j=1}^{p} Z_j W_{jk} \qquad (3)$$

To compute the output signal,

$$Y_k = f(Y_{ink}) \qquad (4)$$

The backward pass:

The error correction term of the output unit,

$$\delta k = (T_k - Y_k) f'(Y_{ink}) \qquad (5)$$

Updating the weights and bias of the output unit,

$$\Delta W_{jk} = \alpha \delta_k Z_j \qquad (6)$$

$$\Delta W_{0k} = \alpha \delta_k \qquad (7)$$

Error correction term of the hidden unit,

$$\delta_{inj} = \sum_{k=1}^{m} \delta_k W_{jk} \quad (8)$$

$$\delta_j = \delta_{inj} \quad f'(Z_{inj}) \qquad (9)$$

Update the weight and bias of the hidden unit,

$$\Delta V_{ij} = \alpha \delta_j x_i \qquad (10)$$

$$\Delta V_{0j} = \alpha \delta_j \qquad (11)$$

Where i is the number of input vectors

j is the number of hidden vectors

k is the number of output vectors

T is the target unit

3.2 Neural Network Design

First we make a neural net structure. This is implemented using MATLAB environment. There are 8 neurons in the input layer and 1 neuron in the output layer. The 8 inputs are "1" and "0" combinations from 8 sensor according to the obstacle detection. Output represents the robot obstacle avoidance angle. After we try several algorithms with much iteration we decide that 16 neurons in the hidden layer. The activation function used in the hidden layer is tan sigmoid and purelin in the output layer. The neural net is shown as in Fig 4.

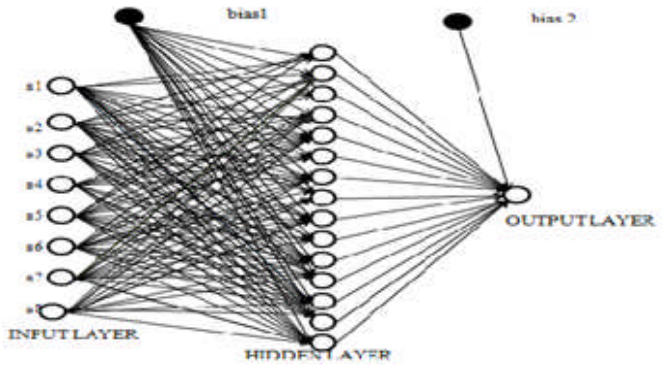


Fig.4. Neural Network Structure

The offline training is done using MATLAB. The performance of the net is evaluated by using Mean Square Error (MSE). This is shown in Fig 5.

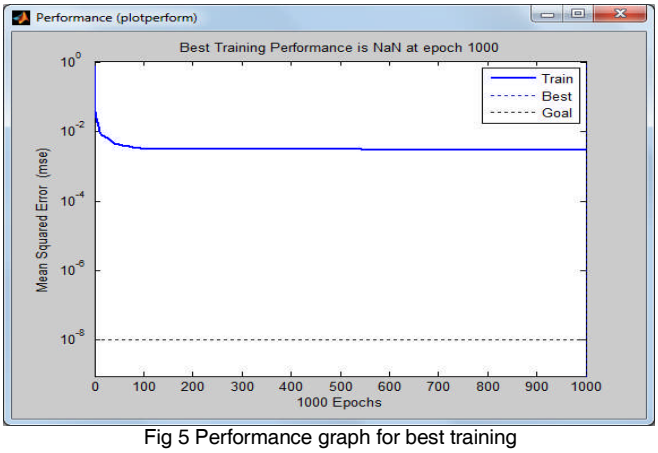


Fig 5 Performance graph for best training

The neural structure is finalized with the 256 sensor input patterns and the corresponding robot target angle. Table 1 shows the some of the training results. Fig 6 shows the actual neural network output Vs target output

Table 1

Pattern of neural network

S.NO	INPUT PATTERN SENSOR OUTPUT "0" & "1"								OUTPUT PATTERN ROBOT TURNING ANGLE (UNIT:DEGREE)
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	1	0
3	0	0	0	0	0	0	1	0	0
4	0	0	0	0	0	0	1	1	0
5	0	0	0	0	0	1	0	0	0
6	0	0	0	0	0	1	0	1	0
7	0	0	0	0	0	1	1	0	0
8	0	0	0	0	0	1	1	1	0
9	0	0	0	0	1	0	0	0	50
10	0	0	0	0	1	0	0	1	50
11	0	0	0	0	1	0	1	0	-45
12	0	0	0	0	1	0	1	1	-45
13	0	0	0	0	1	1	0	0	70

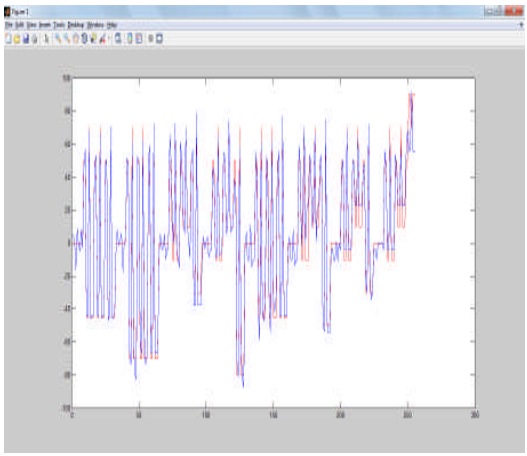


Fig 6 Actual NN output and desired output

3.3 Neural Network Implementation

The IR sensor inputs are interfaced with LabVIEW for final computation. Adjusted weights and bias are transferred. We build a user interface or front panel with controls like knobs, push buttons and toggle switch for sensors and robot turning angle. This is shown in fig 7.

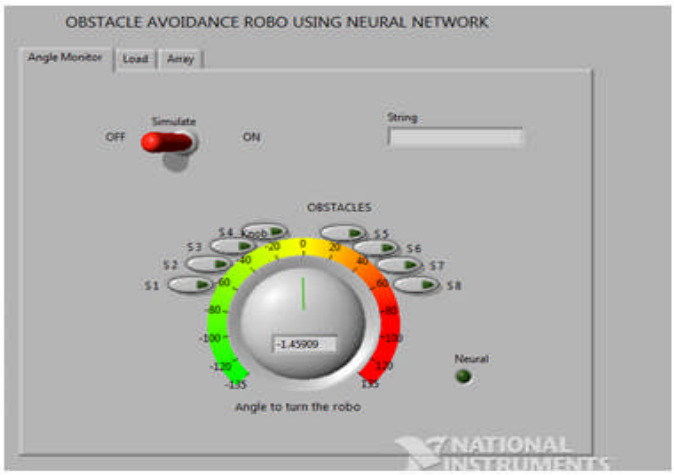


Fig.7. VI implemented for sensors arrangement and robot turning Angle

The real sensor input pattern is compared with already trained input pattern. It is matched with one of the input pattern among 256 combinations. Push buttons which represents virtually as the sensors input are activated according to the input pattern. For the corresponding combination there is a target angle where there is no obstacle. The robot turning angle is shown virtually using LabVIEW by rotating the knob. The block diagram is shown in fig 8.

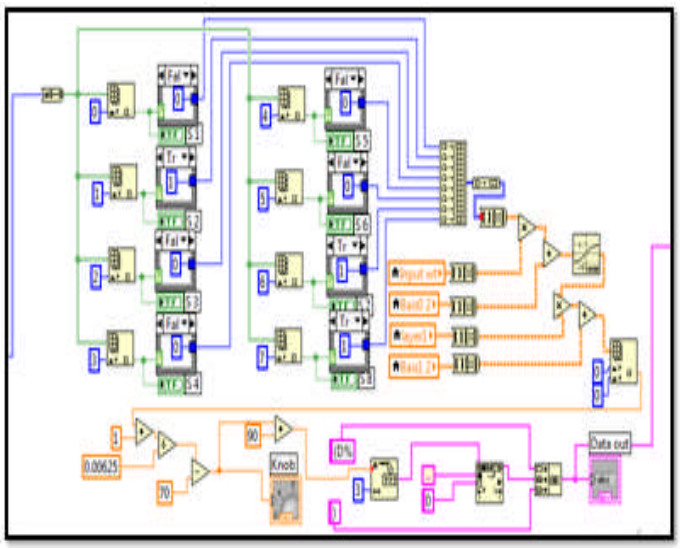


Fig.8. Block diagram window

4 Conclusion

The experimental setup is installed with an intelligent control system by applying the artificial neural network. This NN acquire IR sensor data and learn the environment. This establishes the collision free trajectory for the robot. It is very simple and efficient to find the robot turning angle. The algorithm used for this approach is very simpler and minimize the error considerably. Also it is cost effective.

References

[1] Matt Knudson, Kagan Tumer, (2011) "Adaptive navigation for Robots", Robotics and Autonomous Systems 59 (2011) 410–420

[2] Antonio Sgorbissa,, Renato Zaccaria (2012) "Planning and obstacle avoidance in mobile robotics", Robotics and Autonomous Systems 60 (2012) 628–638

[3] Ehsan Hashemia, Maani Ghaffari Jadidi , Navid Ghaffari Jadidi "Model-based PI–fuzzy control of four-wheeled Omni-directional mobile robots", Robotics and Autonomous Systems 59 (2011) 930–942

[4] Kai-Hui Chi , Min-Fan Ricky Lee (2011) "Obstacle Avoidance in Mobile Robot using neural network", 978-1-61284-459-6/11/ IEEE

[5] Beom, H. R. and H. S. Cho (1992). "A Sensor-based Obstacle Avoidance Controller For A Mobile Robot Using Fuzzy Logic And Neural Network." Intelligent Robots and Systems, 1992, Proceedings Of the 1992 IEEE/RSJ International Conference

[6] Ganapathy, V., Y. Soh Chin, et al. (2009). "Fuzzy and Neural controllersfor acute obstacle avoidance in mobile robot navigation." Advanced Intelligent Mechatronics, 2009. AIM 2009. IEEE/ASME International conference.

[7] Demirli, K. and I. B. T÷rksen (2000). "Sonar based mobile robot Localization by using fuzzy triangulation." Robotics and Autonomous Systems 33(2-3): 109-123.

[8] Harb, M., R. Abielmona, et al. (2009). "Speed control of a mobile robot Using neural networks and fuzzy logic." Neural Networks, 2009. IJCNN 2009. International Joint Conference.

[9] Yau-Zen, C., H. Ren-Ping, et al. (2007). "A Simple Fuzzy Motion Planning Strategy for Autonomous Mobile Robots." Industrial Electronics Society, 2007. IECON 2007. 33rd Annual Conference of the IEEE.

[10] Lin, H. H. and C. C. Tsai (2008). "Laser pose estimation and tracking Using fuzzy extended information filtering for an autonomous Mobile robot." Journal of Intelligent and Robotic Systems: Theory and Applications 53(2): 119-143.