Neural Control System in Obstacle Avoidance in Mobile Robots Using Ultrasonic Sensors

A. Medina-Santiago*[1], J. L. Camas-Anzueto[1], J. A. Vazquez-Feijoo[1, 2], H. R. Hernández-de León[1], R. Mota-Grajales[1]

[1] Instituto Tecnológico de Tuxtla Gutiérrez
División de Posgrado e Investigación
Tuxtla Gutiérrez, Chiapas., México
*amedina@ittg.edu.mx
[2] Centro Interdisciplinario de Investigación para el Desarrollo Integral Regional
Unidad Oaxaca Grupo de Investigación Ingeniería
Oaxaca, Oaxaca., México

ABSTRACT

This paper presents the development and implementation of neural control systems in mobile robots in obstacle avoidance in real time using ultrasonic sensors with complex strategies of decision-making in development (Matlab and Processing). An Arduino embedded platform is used to implement the neural control for field results.

Keywords: neural control system, real-time implementation, navigation environment, obstacle avoidance in mobile robots.

RESUMEN

Este trabajo presenta el desarrollo e implementación de sistemas neuronales de control en robots móviles en evasión de obstáculos en tiempo real utilizando sensores ultrasónicos, con estrategias complejas de toma de decisiones en el desarrollo (Matlab y Processing). Se utiliza plataforma embebida Arduino en la implementación del control neuronal para resultados de campo.

1. Introduction

The navigation of a mobile robot is one of the most important challenges in the field of mobile robotics as these kinds of robots must be able to evade the obstacles they encounter on their way towards a goal. A large number of researchers focus their studies on control techniques and intelligent vehicle navigation, because conventional monitoring techniques are limited due to the uncertainty of the environment where the vehicle is intended to move. Therefore, the need to develop intelligent control strategies, such as neural networks because they offer a very good solution to the problem of navigation of vehicles, their ability to learn nonlinear relationships between the input values and sensor values output.

In short, the problem is a vehicle located in a particular environment, this environment contains a number of random obstacles and a goal. We can pose several ways to solve this problem, on one hand, a classical methods may arise such as route

planning or visibility graphs. On the other hand, we have the option of using an intelligent control system based on artificial intelligence techniques such as neural networks, fuzzy control, etc. These systems are able to solve many control problems due to its ability to emulate some human characteristics.

In [1], a collision-free path between the source and the destination is constructed based on neural networks for mobile robot navigation in partially structured environments. The proposed scheme uses two neural networks for the task. The neural network multilayer perceptron (MLP) using back-propagation. The proposed scheme is carried out in real time implemented on an Intel Pentium 350 MHz processor; the robot is able to avoid all obstacles to reach its goal from the initial starting point.

In [3], the problem of navigation of a mobile robot is solved with the aid of a model of local neural network. This network is a set of sub-models which

represents the dynamic system modelling in various operating points. Each sub-model is a feedforward neural network trained with back-propagation algorithm. The output of these submodels is weighted with the help of a neural network with a base function to generate motion commands for the robot. The performance of the local neural network is compared to a multilayer perceptron network and a radial basis function as a parameter to measure performance, measuring takes the time it takes the robot to reach its goal or destination.

The controller for navigation consists primarily of a set of three subnets neural designs (see 4 and 5). The first two are responsible for the most important behaviours of an intelligent vehicle, which are the location of the target and obstacle avoidance. Both controllers are classifiers, which are trained with backpropagation-supervised techniques. The third neural network acts as a supervisor and is responsible for the final decision based on the outputs of the first two networks. This driver or neural network is trained by the algorithm variant of the associative reward-penalty for learning. Due to its hierarchical structure, system complexity is reduced, resulting in a fast response time.

2. Methodology

This section of the article describes the robot structure analysis, engine capabilities used, housing the brain neural networks sensor system, the system block diagram and the design of the neural network designed and implemented.

2.1 Mobile robot construction

The design of the mobile robot was made to be easily modified and adaptable to new and future research. The physical appearance of the robot was evaluated, and the design was defined based on criteria of functionality, available materials, and mobility. The analysis of different structures of guided robots at a reduced size and simplicity in structure (as shown in Figure 1), another factor considered is the previous working experience with mechanical structures for robots.

From Figure 1, we have:

1. Rear wheels.

2. Geared.

3. Base structure made of 5mm Zintra.

4. Control board formed by the Arduino board.

5. L293B circuit for driving the motors.

6. Idler to permit rotation of the robot.

7. SRF02 Sensor Left
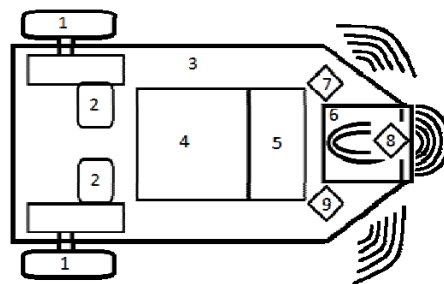
8. Central SRF02 Sensor

9. SRF02 Sensor

Figure 1. Mechanical structure of robot, top view

2.2 Embedded architecture, proximity sensors and geared

The system used in the robot motors robot to give it movement are Geared model number: B02_1-180.

The main features are:

• Gearmotor CD

• Voltage $5Vdc$

• Torque $3\,KgF * cm$

• $43\,RPM$ speed

• No-load current consumption: $75mA$

• Stuck consumption current: $670mA$

• Output shaft $5mm$ diameter

• Holes for screw mounting.

• Measurements: $65mm\,x\,22mm\,x\,18.5mm$. Weight: $32gr$

Figure 2 shows the motor used in the design of mobile robot.

Figure 2. Gearmotor B02_1-180

The embedded system development platform will have the decision-making system based on an artificial neural network with the following characteristics:

- Microcontrolador ATmega368

- Operating voltage: $5Vdc$

- Input voltage: $7 - 12Vdc$

- Pines E / S digital: 14

- Analog input pines

- Current per pin: $40mA$

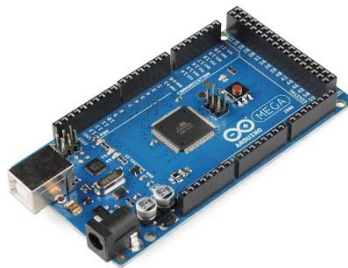- Current at $3.3V$ pin: $50mA$

- Clocked $16MHz$

Figure 3 shows the card used in the mobile robot.

Figure 3. RNA implementation embedded system for making decisions

The ultrasonic detection of obstacles used in the mobile robot, model SRF02 sensors has the following characteristics:

- Range of $15 - 600cm$

- 5V supply voltage.

- Current consumption of $4mA$

- Frequency $40KHz$

- TTL level.

- Dimensions 24x20x17mm device.

- Communication via serial and I2C protocol.

- Low cost.

Figure 4 shows the sensor used for obstacle detection.

Figure 4. SRF02 ultrasonic sensor model

The diagram of connections general designed and implemented in the mobile robot, which contains the embedded system, SRF02 sensors, geared B02_1-180 and LCD screen to show the status of sensors and motors shown in Figure 5.
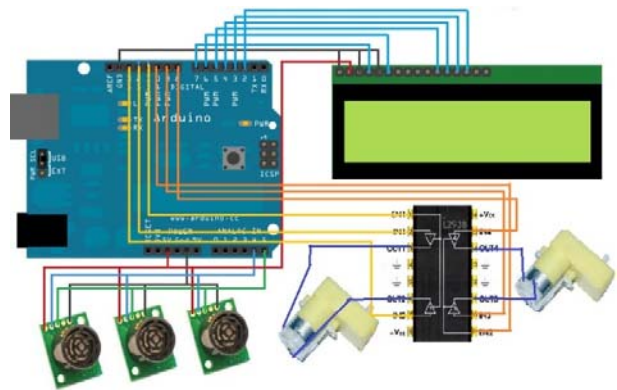

Figure 5. Diagram of connections of general of mobile robot

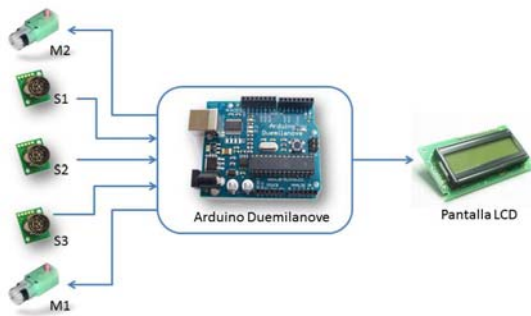In Figure 6, the block diagram of the system operation is shown.

Figure 6. Diagram block of the system operation

2.3 Design of an artificial neural network for decision-making

Controlling the movements of mobile robot that must generate a path that allows itself to navigate avoiding obstacles without colliding, was identified as a problem of pattern classification. For such problems we can find networks as single perceptron, multilayer perceptron, the backpropagation networks, radial basis networks and probabilistic neural networks (PNN) among others, as the most commonly used in this type of application.

The neural network chosen for the development of robot control system is a multi-layer perceptron (MLP), because such networks have good characteristics to solve classification problems as the present one. Furthermore, we can add that it presents a relatively simple architecture for implementation in the robot.

Figure 7 shows the structure of the network used a neural network of three inputs a vector output of 4 elements, with a 4-layer configuration was used and [5, 4, 4, 4], in the first hidden layer with activation function type " tansig " in the second hidden layer and the third layer with a hidden function type "logsig " as the output layer activation function with a "logsig ".

The inputs to the network are the information from sensors placed on the left, center and right, and the sensors are activated according to the proximity of an obstacle in that guidance. The network will be output according to this post, which will allow correcting the position and thus, avoiding the obstacle. As we know the output function of the network makes the output values are decimal numbers in the range [0 1], then

we program a simple comparison function within the Arduino to set the output value greater than 1 and the other two to 0.
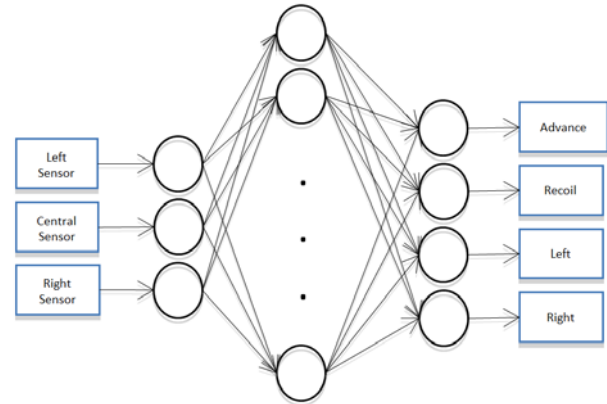

Figure 7. Structure of the network using a neural network of three inputs and a vector output of 4 elements

3. Results

The output vector consists of three elements corresponding to each of the basic activities that can control the robot. The values of these elements are coded, so that the corresponding movement to run es1 and the other three to 0. Thus, the network will provide the response vector (1,0,0,0) if the control action is to turn right, (0,1,0,0) if you turn left, (0,0,1, 0) if the action is to advance and finally (0,0,0,1) if the action is back.

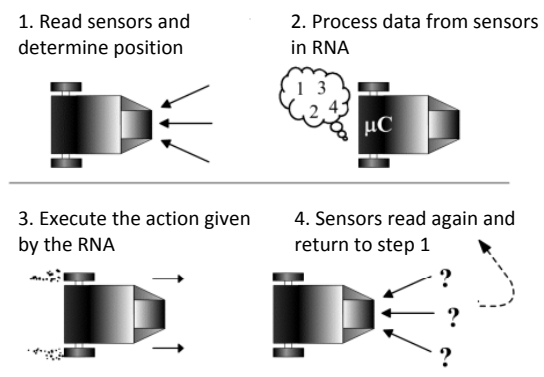The algorithm developed is shown in Figure 8.

Figure 8. Algorithm for decision-making using RNA-BP

The Neural algorithm developed and implemented in the mobile robot was programmed in C language

for implementation on embedded system. The neural network architecture and algorithm design Apprenticeship is monitored and offline in Matlab, where the values of synaptic weights and Bias extract entailment respectively. The parameters of the neural network training:

S1 = 18;
[R,Q] = size(P1);
[S2,Q] = size(targets);

net = newff(minmax(P1),[5,4,4,S2],{'tansig','logsig', 'logsig','logsig'},'trainrp');

net.performFcn = 'sse';
net.trainParam.goal = 0.1;
net.trainParam.show = 20;
net.trainParam.epochs = 6000;
net.trainParam.mc = 0.95;
[net,tr] = train(net,P1,targets);

R = sim(net,P1);

Figure 9 shows the performance characteristics of the neural network in the training backpropagation Matlab shown satisfactory results observed training.

Figure 9. Performance characteristics of the neural network in the training backprogation

Figure 10 shows the behavior of the error, decreasing during the training of the RNA-BP.
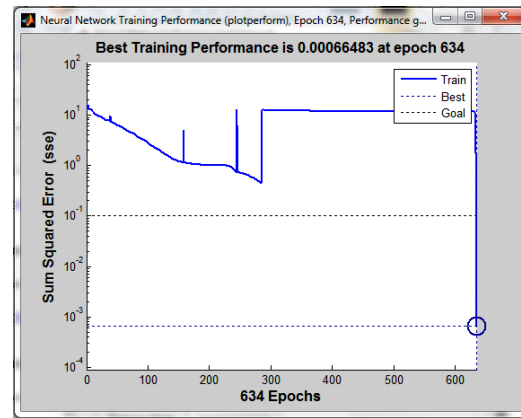

Figure 10. Neural network training performance

Programming the BP neural network developed and implemented for mobile robot in the embedded system depends mainly on:

• Synaptic weights of the hidden layers

• Synaptic weights of the output layer

• Bias or king pesos each layer

• Know the activation functions used in each of the layers. In the first hidden layer is used tansig and the remaining was used logsig

For the weights of the input layer is made using the command: "net.IW {}", and the weights of the remaining layers with the command: "net.LW {}", finally the bias values are obtained with: "net.b {}". These data and bias weights are stored in the memory of the embedded system in matrix form, to be applied to the verification formulas MLP network which are:

$$n_j^o = \sum_{i=1}^{q} W_{ji}^o P_i + b_j^o \qquad (1)$$

W^o_{ji} = Weight connecting the input layer to the hidden layer

P_i = Component of the input vector

b^o_j = Gain of the hidden layer, extra weights in the networks

This formula multiplies each input with its own weight and then sums the corresponding bias for each neuron. Each of the neurons of the first hidden layer has an output given by:

$$a_i = \tan sig \left(\sum_j w_{ji} P_i - b^o_{\ j} \right) = \frac{e^{n^o_{\ j}} + e^{-n^o_{\ j}}}{e^{n^o_{\ j}} - e^{-n^o_{\ j}}} \qquad (2)$$

The remaining neurons of the remaining layers have an activation function given by the following formula:

$$n_k = \log sig \left(\sum_j s_i w_{ij_i} + b^o_{\ j} \right) = \frac{1}{1 + e^{-sj}} \qquad (3)$$

The weights connecting the input layer to the hidden layer are stored in a w1 matrix with five rows and three columns, variables inputs are acquired and stored in p [ ] vector, the weights connecting the hidden layer with the next layer are stored in the array w2 four rows and five columns, the weights connecting the next layer to the next are stored in the array w3 four rows and four columns, as with the last hidden layer and the layer output are stored in the array w4 four rows and four columns. The bias corresponding to each layer which are stored in a vector, for a total of four vectors, b1 [5], b2 [4], b3 [4], b4 [4]. The output of this network decides the action to follow hence you can avoid the obstacles encountered in the environment.

Thanks to the accuracy of the generated C functions for working with embedded floating point output of the network system is equivalent to the ones processed by Matlab. Being successful the integration of the neural network to the embedded system.

The code for generating the activation function and logsig tansig C is:

a1[i] = 2/(1+pow(2.71828,-2*n1[i]))-1   (4)

a2[i] = 1/(1+pow(2.71828,-(n2[i])))   (5)

4. Physical implementation and analysis

The final robot's dimensions shown in Figures 9 and 10, are 26 cm long, 12 cm tall, 24cm wheel separation and an average speed of 10cm/s travelling in a straight line. The wheels are covered with non-slip material to improve performance on smooth surfaces. The sensors work in an acceptable manner, as sensors for these objects must be relatively dense material for the ultrasonic signal to be reflected achieved successfully and detected by the transducer.

Full navigation and learning algorithm was implemented in no more than 200 lines of code in C language and is easily extensible to include new situations and actions.

The adaptability of the robot was tasted by having the robot to interact with a group of people in an open exhibition space, which play as dynamic obstacles, i.e., used their feet to block the movement of the robot, I've even another mobile robot.
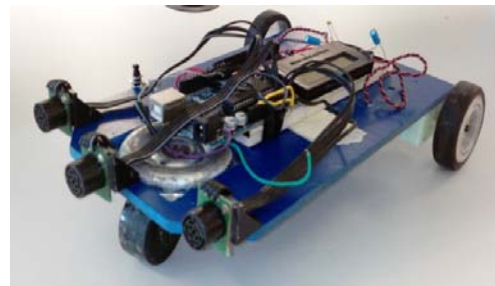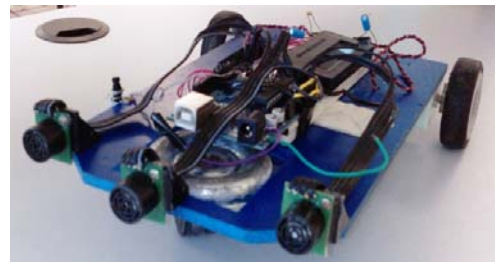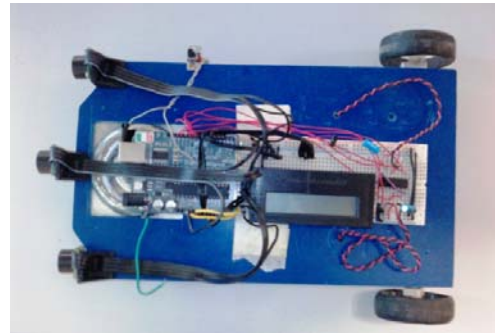

Figure 11. Experimental physical design of mobile robot based on neural network with backpropagation algorithm

5. Conclusions

Neural networks are excellent tools applicable in mobile robots evading obstacles, which have the ability to work with imprecise information

The architecture of the RNA-BP is implemented in embedded processing capabilities in a real time system, generating response cases, which were not taken into account when training, happening to be a reliable and fault-tolerant control. The vehicle control resulting in robust differential drive system that was used.

The SRF02 sensors that are used are suitable for this application; however, further research is suggested to integrate a SRF08 series that offers better supervision, ease calibration, and size than the SRF02 series. More needs to be achieved though: Reliability of implementation and management of the back-propagation neural network design.

For more advanced applications is recommended to work with a DSP. The use of techniques that involve more time together sensors, sensors such as gyroscopes, electronic compass, and other encoder to guarantee the position of the movable in space is suggested. Using more proximity sensors, at least five or six will give us a better understanding of the environment as long as the DSP support computational burden of neural network that would be generated.

Acknowledgements

References

[1] Danica Janglova, "Neural networks in mobile robot motion," International Journal of Advanced Robotic System, vol. 1, no. 1, 2004, pp. 15-22.

[2] Jason A. Janet, Ricardo Gutiérrez, Troy A. Chase, Mark W. White, and John C. Sutton, III, "Autonomous mobile robot global self localization using kohonen and region-feature neural networks," Journal of Robotic Systems, 14 (4), 1997, pp. 263-282.

[3] Hamdi A. Awad, and Mohamed A. Al-Zorkany, "Mobile robot navigation using local model networks," International Journal of Information Technology, vol. 1, no. 2, pp. 58-63.

[4] A. Chohra, A. Farah, and C. Benmehrez, "Neural navigation approach for intelligent autonomous vehicles in partially structured environments," Applied Intelligence, vol. 8, no. 3, May-June 1998.

[5] Brooks, R "A Robust Layered Control System for a Mobile Robot" IEEE Journal or Robotics and Automation, Vol. RA-2

[6] Martin T. Hagan, Howard B. Demuth, Mark Beale "Neural Networks Design".

[7] A Practical Guided to Neural Nets, Marilyn McCordNelso, W. T. Illingworth. Texas Instrument. Addison-Wesley Publishing Company, Inc.

[8] Neural and Adaptive Systems Fundamentals Through Simulations, José C. Principe, Neil R. Euliano, W. CurtLefbvre, John Wiley&Sons, Inc.

[9] Raul Rojas, Neural Networks A Systematic Introduction, Springer, 1996.