

# A RAM-Based Neural Network for Collision Avoidance in a Mobile Robot

Qiang Yao, Daryl Beetner, Donald C. Wunsch II and Björn Osterloh\*

Department of Electrical & Computer Engineering,  
University of Missouri-Rolla, Rolla, MO, 65401

\* Institut fuer Datentechnik und Kommunikationsnetze,  
Technische Universitaet Braunschweig, Braunschweig, Germany

**Abstract**—A RAM-based neural network is being developed for a mobile robot controlled by a simple microprocessor system. Conventional neural networks often require a powerful and sophisticated computer system. Training a multi-layer neural network requires repeated presentation of training data, which often results in very long learning time. The goal for this paper is to demonstrate that RAM-based neural networks are a suitable choice for embedded applications with few computational resources. This functionality is demonstrated in a simple robot powered by an 8051 microcontroller with 512 bytes of RAM. The RAM-based neural network allows the robot to detect and avoid obstacles in real time.

**Keywords:** RAM-based neural network, robot navigation, microprocessor system.

## I. INTRODUCTION

Research and development of autonomous mobile behaviors has a long history. Some of the capabilities that have been developed include road-following, adaptive vehicle speed control, and obstacle detection and avoidance [1]. The road following and speed control technologies are fairly mature and reliable. The obstacle detection and avoidance capability is much less reliable and, therefore, requires further research and development.

This research concerns development of efficient neural networks for real-time detection and avoidance of obstacles in a mobile robot based on a simple microprocessor system. Conventional neural networks often require powerful computer systems to implement in a reasonable time-frame, and require a time-consuming training process. Neural networks that can be implemented with relatively modest computer hardware could be very useful. Some work done by others indicates this approach may be successful. Mitchell *et al* [2] has shown that a simple Hopfield network can be used to control an insect robot's movement, which can move forwards, backwards or move around an obstacle, by using only a Z80-based microcontroller system. In an unpublished paper by Zhou *et al* [3], a simple perceptron network was developed to control a fire-fighting robot. The neural network was used to detect obstacles, find a fire, and then the robot moved to extinguish the fire. In both papers, the neural networks had limited power. For example, the fire-fighting robot was likely to oscillate between detecting objects to the left and right. Its network learning made little progress in this case.

In this paper, a RAM-based neural network is presented for use in an autonomous robot. The robot used for this study was originally developed to navigate through a maze by students of the Department of Electrical and Computer Engineering at the University of Missouri-Rolla. This robot is used in our research to demonstrate the capability of neural networks on simple computing platforms. In our work, the robot has to move forward from a starting position, detect any

obstacles in its way, and navigate around objects. Object type and location are unknown to the robot. A RAM-based neural network is used for this task since RAM-based neural networks use arithmetic and logic functions instead of complex floating point operations and can readily and efficiently be implemented in hardware. These networks are typically used in image-recognition applications because of their small size and computational requirements [4][5]. It should be noted that, while neural networks may not necessarily be required to perform the tasks implemented in the robot up to this point, this work demonstrates the potential of neural networks in embedded applications where powerful computers may not be available.

In the paragraphs that follow, the robot's mechanical and sensor system and the RAM-based neural network structure will be presented. Results of experiments that measure the robot's collision-avoidance ability will also be given.

## II. THE ROBOT

The robot is a two-wheeled device that moves and turns using two separate stepper motors, as illustrated in figure 1. It consists of an 8051-based microprocessor with 512 bytes of RAM, 32K flash memory, and an RS 232 port which is used for in-system programming of the device. Processor code was developed using Keil C51, which is a special subset of C language for the 8051 microcontroller [6].

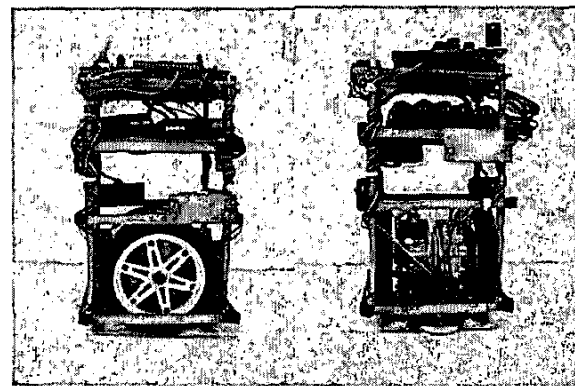


Fig 1. The robot (front and side view)

Three infrared (IR) sensors (Sharp GP2D02) are mounted on the robot, which enable the robot to measure the distance between it and an object directly ahead of it or to its right and left side, as shown in figure 2. Each IR sensor outputs an 8-bit binary number to describe a distance between 10 cm and 80 cm to an object. The detection angle is about 12°. The short and narrow range of the infrared sensors and their orientation (figure 2) prevent detection of peripheral objects. As a result, the robot may become stuck in a corner or collide

with obstacles. In order to overcome this problem, the robot "sweeps" its environment with the front detecting sensor over an angle of about 60 degree to get additional range data, as shown in figure 3 [7].

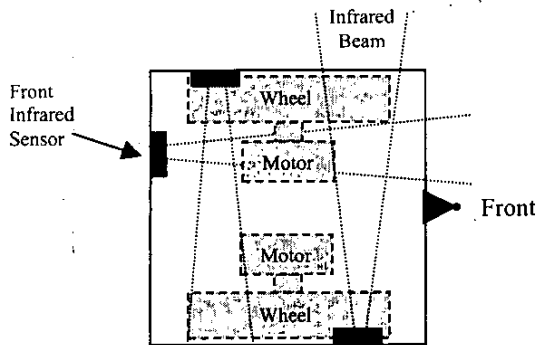


Fig 2. Sensor placement

The robot rotates approximately  $3.5^\circ$  about its center when its two wheels are each turned one step in different directions. The robot is programmed to turn from the left to right in 16 discrete steps, with the microcontroller reading range information from the front sensor every four steps. Five range values are obtained after a scan, which describes the robot's surroundings in the right, the right-front, the front, the left-front and the left directions. This range information is used as input for the RAM-based neural network.

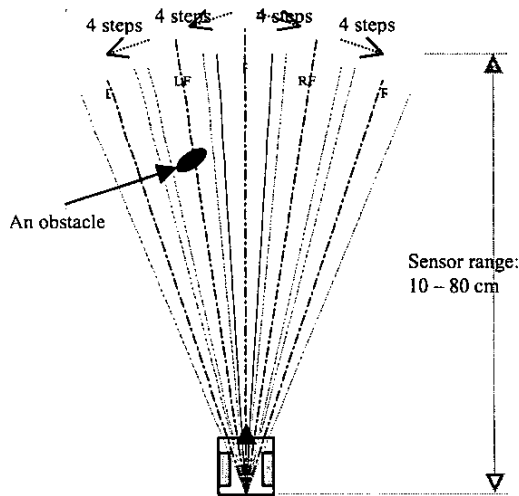


Fig 3. Scanning with the IR sensors

### III. NEURAL NETWORK FOR OBSTACLE AVOIDANCE

#### 1. RAM-Based Neural Network

RAM is the most cost effective digital component and has some properties in common with a McCulloch and Pitts (MCP) neuron. The network can be trained by writing to memory and the trained data can then be recalled (read).

Obviously, training in this manner is simpler than modifying weights with the delta rule [5].

A RAM discriminator is a structure made up of multiple RAM-nodes. Several discriminators are used in a group that is called a *classifier* or *neuron* [5][8]. Each discriminator is structurally identical, but is trained with different patterns. A classifier has the same properties as a single-layer perceptron. Network output comes from calculating the sum of the activated neurons in each class. Training is by example, which is very fast as it is a one-step process. Generalization allows an adequately trained network to identify a previously unseen input.

#### 2. The Robot's Network

Data from a scan is split into five directions. In each direction, an obstacle may or may not appear. The combination of obstacles' appearance in the five different directions makes up different input patterns. The RAM-based neural network is designed to identify the current environment by recognizing typical patterns, which were encountered during training of the neural network. Therefore, the obstacle avoidance problem is transformed into a pattern-recognition problem, which can be easily solved.

A two-layer RAM-based neural network was used in our work. The structure is shown in figure 4.

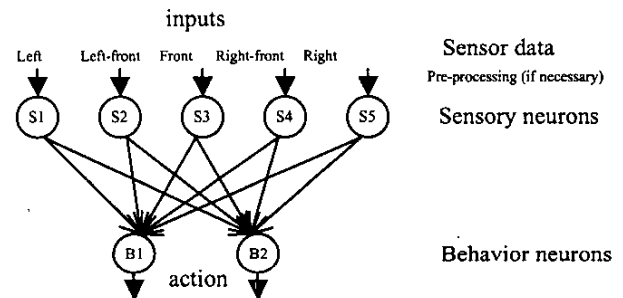


Fig 4. The RAM-based neural network

To implement the RAM-based neural network, the 8-bit data for each sensor was split into two halves, the upper and lower 4 bits. The five bytes of sensor data thus produce 10 groups of 4-tuple data.

#### Sensor Layer:

The sensor layer, also the first layer, identifies where obstacles may lie in five discrete directions including the Right, Right-Front, Front, Left-Front and Left directions. Three cases are possible for each direction, including no obstacle (00), far obstacle (01) and near obstacle (10). There are a total of five neurons in this layer. Each neuron includes two discriminators. Because of symmetry, the left neuron uses the same discriminators as the right neuron. The same is true for the left-front neuron and the right-front neuron. A total of six different discriminators are used. They are:

- Discriminator11 and Discriminator12: they determine obstacles in the right or left direction.
- Discriminator13 and Discriminator14: they determine obstacles in the right-front or left-front direction.

- Discriminator15 and Discriminator16: they determine obstacles in the front (forward) direction.

Each neuron has the structure shown diagrammatically in figure 5. Each neuron used in the model has the same internal structure, although they are functionally different. Two discriminators are used to identify if an obstacle exists or if an obstacle is far away or near. The discriminator registers are a set of RAM cells, which store the weights describing a pre-learned pattern. Those weights are saved during the training process. The 4-bit input data are used as an address to look up weights for both discriminators. The 4-bit input accesses the corresponding register cell and gets the weight. The sum of the weights is then calculated and compared with pre-set thresholds. If the result is equal to the far threshold, the output is 01; if the result is equal to the near threshold, the output is 10. Otherwise the output is 00 (no object).

We will introduce how to train the neural network later.

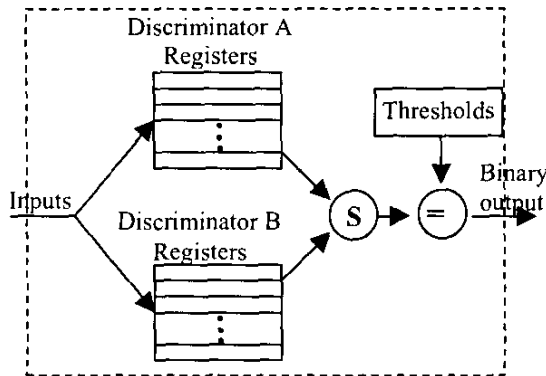


Fig 5. Inside a neuron

### Behavior Layer:

The behavior layer is the second layer, which identifies the whole environment based on the outputs of the first layer. Namely, it combines the obstacles identified in the five directions together and then makes a suitable avoidance decision. Two neurons are used here. Each includes two discriminators and is similar in structure to that shown in figure 5, except that the sum operation is substituted by an OR operation. The inputs are the five two-bit data outputs from the sensor layer. Those data are divided into four groups of behavior layer inputs, as shown in figure 6. The high five bits are divided into a left class C1B1A1 and right class C1D1E1, which are sent into the first neuron. The low five bits are divided into a left class C0B0A0 and right class C0D0E0, which are sent into the second neuron. In the neuron, these bits are used as addresses to the discriminator registers and to select appropriate weights. The outputs of both neurons are compared to choose a final action. The C0 and C1 bits are used twice in each discriminator. Their dual use does not affect the neuron's function. The discriminators just use more RAM.

Based on the output of the second layer, one of nine actions is taken:

- Go forward
- Turn 90° to the left or right,
- Turn 67.5° to the left or right

- Turn 45° to the left or right,
- Turn 22.5° to the left or right

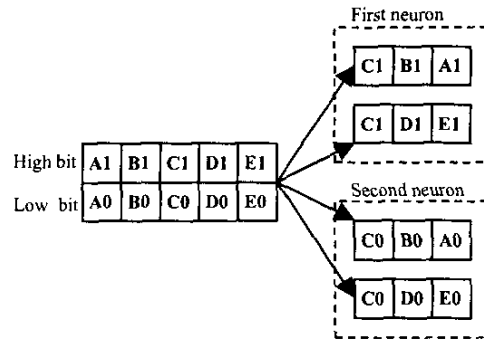


Fig 6. Inputs to the behavior layer

The relative position between the robot and the obstacle determine which action is selected. For example, if a single obstacle is located far away and to the left, the robot will turn 22.5° to the right. If a single obstacle is located far away and to the left-front, the robot will turn 45° to the right. If there is more than one obstacle, the network can determine if the gap between the two objects is wide enough for the robot to pass through. If not, the network will make the robot turn 90° to the left or right to avoid the obstacle combination.

### Training the RAM-based NN

Because a RAM-based network has no weight matrix and cannot be trained using a back-propagation like algorithm, a reinforcement learning method has been used. Reinforcement learning does not require a direct measure of error, but just an indication that the output was incorrect [5]. Here we use patterns with a single far obstacle or with a single near obstacle to train the neural network. At a time the obstacle is placed in different directions and in different distance. The neural network then is taught that the obstacle is far or near, at left or at right. Using this technique, the value distinguishing a far obstacle and a near obstacle has to be obtained first.

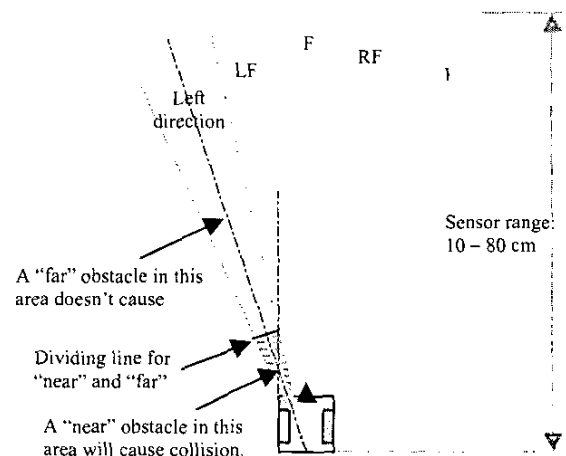


Fig 7. Choosing the training values

The calculation of this value is based on the distance from an obstacle to the robot, the sensor's detection angle, and the size of the robot, as shown in figure 7. Based on this calculation, the threshold values for the robot are 0111,1111 (about 45 cm) for the right-front and left-front directions and 1100,1111 (about 24 cm) for the right and left directions. Taking into account the sensor non-linearity and measuring error, we also define 0000,1111 (75 cm) as the threshold indicating that no obstacle is detected. That is, any sensor value less than 0000,1111 is viewed as indicating that no obstacle exists.

Figure 8 shows an example of discriminator 1 and 2, which are used in the left direction neuron. The calculated pattern has been saved in the RAM cells of each discriminator during the off-line training. The RAM-based neural network has the ability to learn with only one presentation of the training data. Therefore the network obtains training with those classification patterns.

Discriminator 1		Discriminator 2	
Inputs	Content	Inputs	Content
0000	00000000	0000	00000000
0001	00010000	0001	00000000
0010	00010000	0010	00000000
0011	00010000	0011	00000000
0100	00010000	0100	00000010
0101	00010000	0101	00000010
0110	00010000	0110	00000010
0111	00010000	0111	00000010
1000	00100000	1000	00000100
1001	00100000	1001	00000100
1010	00100000	1010	00000100
1011	00100000	1011	00000100
1100	00100000	1100	00001000
1101	00100000	1101	00001000
1110	00100000	1110	00001000
1111	00100000	1111	00001000

Fig 8. An example of weights in the trained discriminators

#### IV. EXPERIMENT AND RESULTS

Since the sensor system is very simple, a simple experimental setup was used to test the robot's ability of detecting and avoiding obstacles. A white box was used as the obstacle in our preliminary tests. The box was placed at various random positions. All experiments were conducted on a gray-colored floor.

The robot learned to go forward in the open field when there was no obstacle. When an obstacle was put in its way, it was able to find the obstacle and select an appropriate avoid-

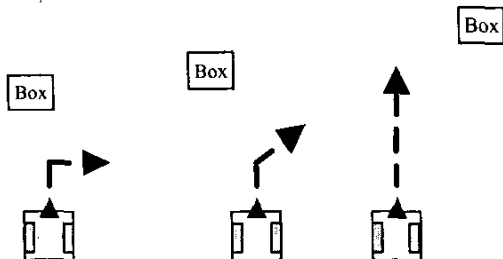


Fig 9. The robot avoids obstacle

ance action. The action depended on the relative position between the obstacle and robot, as shown in figure 9.

The robot was able to detect and avoid obstacles in real-time, even though the robot used a very simple processor and sensor scheme. Since the robot had no destination in mind, it kept moving in a single direction after avoiding an object. A new robot is being developed, which has more infrared sensors (*i.e.* four) to allow better object detection and has 4 phototransistors to allow tracking of a "goal" destination. This robot will be used to further study the use of neural networks on simple computing platforms for the much more complex task of detecting and avoiding objects while attempting to reach a pre-defined destination. An improved RAM-based neural network is being developed to help accomplish this task.

#### V. SUMMARY AND CONCLUSIONS

RAM-based neural networks are mainly used in the area of pattern recognition. In this paper, a new application of RAM-based neural networks in mobile robotics for obstacle detection and avoidance was presented. The network was implemented in a robot with a very modest microprocessor system and a small amount of data memory (512 bytes). The RAM-based neural network worked well even with few computational resources. The network was easy to train and the robot demonstrated the ability to detect obstacles and navigate within its environment in real time. A more sophisticated robot is being developed to improve its ability to detect objects and to show the ability of neural networks implemented on a simple processor platform to perform the difficult task of achieving a destination objective in an environment with unknown obstacles.

#### ACKNOWLEDGEMENTS

This work was supported in part by the National Science Foundation and by the UMR Mary K. Finley endowment.

#### REFERENCES

- [1] O. Omidvar, *Neural System for Robotics*. Academic Press, pp 210-221, 1997
- [2] R. J. Mitchell, D. A. Keating, and C. Kambhampati, "Neural network controller for mobile robot insect," Internal report, Department of Cybernetics, University of Reading, April 1994
- [3] Y. Zhou, D. Wilkins, R. P. Cook, "Neural network for a fire-fighting robot," University of Mississippi
- [4] I. Aleksander, W. V. Thomas, P. A. Bowden, "Wisard: A radical step forward in image recognition." *Sensor Review*, pp 120-124, July 1984
- [5] J. Austin, "RAM-Base Neural Networks", *World Scientific*, pp 8-9, 1998
- [6] Schultz *et al*, "C and the 8051", Prentice Hall, 1998
- [7] B. Osterloh, "RAM-Based Neural Network in Autonomous Mobile Robot," will be published on 3rd ACIS International Conference on Software Engineering, Artificial Intelligence, Networking & Parallel Computing (SNPD '03)
- [8] L. Teresa *et al*, "Weightless Neural Models: A Review of Current and Past Works", *Neural Computer Surveys* 2, pp 41-61, 1999