A Comparison of Optical Modeling and Neural Networks for Robot Guidance

Sameer Parasnis, Sasanka Velidandla, Ernest Hall, and Sam Anand
Center for Robotics Research
Mechanical, Industrial, and Nuclear Engineering,
University of Cincinnati
Cincinnati, OH 45221

ABSTRACT

A truly autonomous robot must sense its environment and react appropriately. These issues attain greater importance in an outdoor, variable environment. Previous mobile robot perception systems have relied on hand-coded algorithms for processing sensor information. Recent techniques involve the use of artificial neural networks to process sensor data for mobile robot guidance. A comparison of a fuzzy logic control for an AGV and a neural network perception is described in this work. A mobile robot test bed has been constructed using a golf cart base. The test bed has a fuzzy logic controller which uses both vision and obstacle information and provides the steering and speed controls to the robot. A feed-forward neural network is described to guide the robot using vision and range data. Suitable criteria for comparison will be formulated and the hand-coded system compared with a connectionist model. A comparison of the two systems, with performance, efficiency and reliability as the criteria, will be achieved. The significance of this work is that it provides comparative tradeoffs on two important robot guidance methods.

Keywords: fuzzy logic, neural networks, autonomous guidance

1. Introduction

Humans receive a large amount of their information through the human vision system, which enables them to adapt quickly to changes in their environment. A truly autonomous robot must sense its environment and react appropriately. These issues attain greater importance in an outdoor, variable environment. The Center for Robotics Research at the University of Cincinnati has been involved in a nation-wide competition to build a small-unmanned autonomous ground vehicle that can navigate around an outdoor obstacle course. Vision-based mobile robot guidance has proven difficult for classical machine vision methods because of the diversity and real time constraints inherent in the task. Various methods have been used before to guide a mobile robot autonomously in an obstacle course. Previous methods have relied upon hand-coded algorithms for processing sensor information. The test bed developed by the University of Cincinnati currently uses the fuzzy logic controller. Recent trends show the use of Artificial Neural Networks to process sensor data for navigation in a complex environment. The purpose of this paper is to compare the techniques of a fuzzy logic controller and the use of a neural network, using performance and efficiency as the criteria.

The paper is organized as follows. In Section 2 and 3, a detailed description of both of the techniques is given. Then the two models are compared in section 4 using suitable criteria. We use some experimental and some simulated data to make the comparison. We present the results in section 5 and follow them with conclusions and further ideas in section 6.

2. The Use of Fuzzy Logic

The specific challenge in the design of an intelligent controller is to know what information is needed, how to measure it and how to use it so as to satisfy the requirements of the system. This involves a mathematical description of the relation among inputs to the process, its state variables and its output. In other words, we build a mathematical model of the system. The modeling of a mobile robot is a very complex task, which has multiple input and multiple outputs, and incorrect modeling can lead to unstable systems and unsuitable performance. The Fuzzy logic Control (FLC) uses the qualitative aspects of the human decision process to construct the control algorithm.

Fuzzy Logic is a way of interfacing inherently analog processes that move through a continuous range to a digital computer that likes to see things as well-defined numeric values. Conventional Boolean logic contains only binary truth-values (0 and 1) where there is a sharp distinction between true and false. Fuzzy logic allows these states to change gradually from one state to the next. In other words, there are values which range between 0 and 1.These are called the membership functions.

The development of techniques for autonomous robot navigation constitutes one of the major trends in the current research in mobile robotics. By autonomous, we mean the ability of the robot to move on its own without human assistance in environments that have not been engineered specifically for them. The design requirements were to build an autonomous robot that would follow a given track separated by two lines and, in turn, avoid the obstacles. This called for a design of separate subsystems at lower levels which would satisfy specific design objectives and integration of these systems at a higher level to enable the robot to function properly.

The subsystems that are designed include a speed control system, a steering system, a vision system and an obstacle avoidance system. In our discussion here, we would primarily focus on the vision system, which consists of 2 CCD cameras and an image-tracking device from ISCAN Inc. The information from the vision system is used as input to a closed loop fuzzy logic controller to control the steering and speed of the robot. Fuzzy controllers are very simple conceptually. They consist of an input stage, a processing stage, and a output stage. The input stage maps sensor or other inputs (in our case the distance and the angle of the line) to the appropriate membership functions and truth values. The processing stage consists of the fuzzy inference engine, which invokes the rules and fires the appropriate response, and then combines the results of the rules. The output stage converts (defuzzifies) the combined result back into a crisp value that is given to the control computer that takes necessary actions.

2.1 The Vision System

In order to follow a track, which is separated by 2 lines, 2 CCD cameras and an image tracking device (ISCAN) are used. The ISCAN image tracking system finds the centroid of the darkest or the brightest region and returns the co-ordinates of these points. These are the image co-ordinates. These co-ordinates are two-dimensional while the real world co-ordinates are three-dimensional. An algorithm is developed to establish a mathematical and geometrical relationship between the physical three-dimensional (3-D) and its corresponding digitized two-dimensional (2-D) co-ordinates [1]. In an autonomous situation the challenge is to determine 3-D co-ordinates given the image co-ordinates. This is established by what is popularly known as “Calibration” of the camera. The objective is to find any corresponding ground co-ordinate given an image co-ordinate. What makes this the most important and crucial task is that the process of following the line is autonomous and dynamic and hence the relationship between these co-ordinates should be accurately determined. This, in turn, determines how closely the robot follows the line and hence the success of the robot.

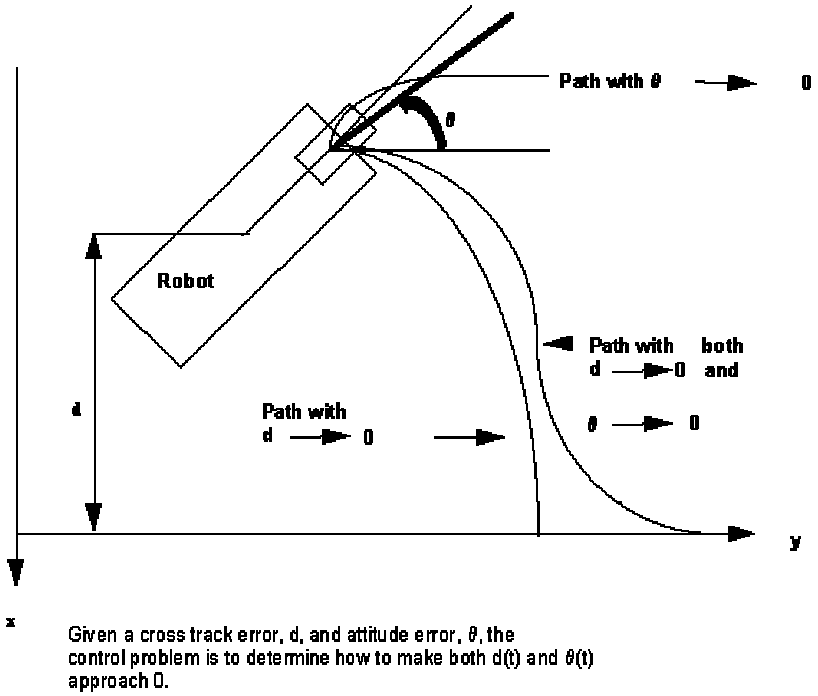


Fig 1.Distance error and angle error of the position of the robot w.r.t the line.

The ISCAN tracking device returns the X and Y co-ordinates of 2 points on the line (Z is constant) and the corresponding ground co-ordinates are calculated as described above. From the computed X and Y co-ordinates of the points the angle of the line with respect to the centroid of the robot is computed from simple trigonometric relationship. The distance of the line from the centroid of the robot is also obtained. Both these parameters need to be controlled. If only the angle is controlled, the robot might draw closer to the line and eventually go out of bounds. The same is true for minimizing only the distance error. By knowing the angle error (θ_{error}) and the distance error (d_{error}), we have to find the steering angle to guide the robot in the proper direction.

2.2 The Fuzzy Logic Controller

The fuzzy control for the autonomous robot was developed based on empirical methods, counting on human experience, chiefly by trial and error. The general process was as follows:

A) Fuzzy Set Definition:

As defined above θ_{error} and d_{error} form the input variables for the controller. Fuzzy subsets are formed on these variables and membership functions are defined. To model the problem in a simple manner, triangular membership functions have been used.

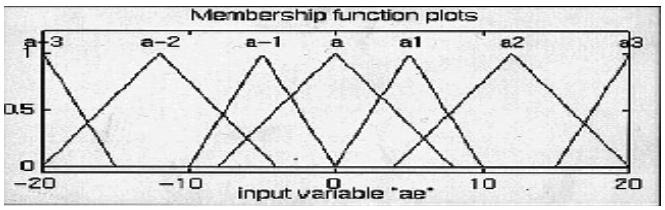


Fig 2.Input Fuzzy Sets (Angle error)

The inputs are then expressed in words describing the degree of membership of the variable.

Fuzzy Set	Description
a-3	Extreme Left
a-2	Left
a-1	Soft left
a	Zero
a1	Soft Right
a2	Right
a3	Extreme Right

Table 1.Lingustic variables for the steering angle.

The output variable i.e. steering angle of the vehicle is similarly fuzzified.

B) Rule Base:

Since navigation as done by humans is an intuitive process, the rule base for this system was derived empirically from extensive testing. Moreover, it was felt that describing navigation in words was a “natural” and “simple” process for the human expert. This description is translated into linguistic rules. Thus a rule base was created. The experimental process yielded a set of 32 rules for guiding the robot along the track and avoiding the obstacles [2].

C) Fuzzy Operators:

Fuzzy Operators are employed to unify the multiple input sets into a single value. The two inputs θ_{error} and d_{error} are combined with the AND operator (product). The resulting value is then applied to the output membership function.

D) Defuzzification:

Centroidal Defuzzification is carried out to obtain the desired steering angle of the robot.

Flow chart for the vision feed back for the controller

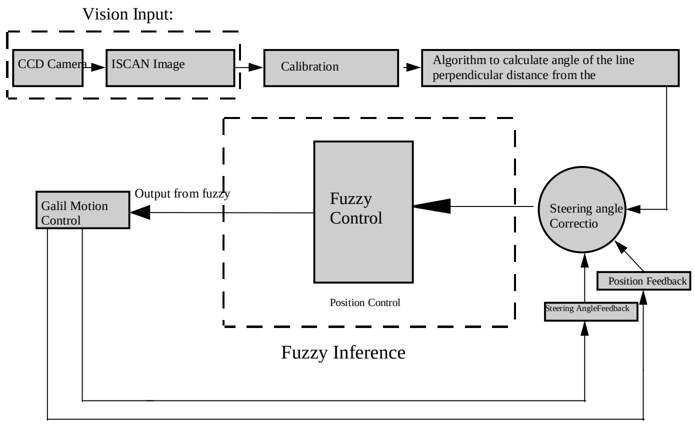


Fig.3

2.3 Obstacle Avoidance

The obstacle avoidance system consists of six ultrasonic transducers. An ultrasonic ranging system from Polaroid is used for calibration of ultrasonic transducers.

The operational principle of the system is that a pulse of electronic sound is transmitted towards the target and the resulting echo is detected. The time between the pulse sent and received back is measured and, knowing the speed of sound in air, the system can convert the elapsed time into a distance.

A fuzzy logic controller to perform the obstacle avoidance was developed as a separate module. The θ_{error} and d_{error} , the distance of the obstacle from the robot, along with the position of the obstacle relative to the robot (LEFT,RIGHT) were the input variables. The desired steering angle is the output. A rule base similar to the one used to avoid obstacles was developed in a manner similar to the one described above. A Pseudo code is developed for obstacle avoidance in the control algorithm, which is then programmed in C and interfaced with the other modules. Testing of this dual level fuzzy system is done. First a theoretical simulation was run using the MATLAB fuzzy logic toolbox [2]. The input used for running these simulations were from theoretical cases. The inputs were θ_{error} and d_{error} . The outputs were $\theta_{steering}$ and the speed of the robot.

3. Neural Network Approach

A neural network is a massively parallel distributed processor that has a natural propensity for storing experiential knowledge and making it available for use. It resembles the brain in two respects:

1. Knowledge is acquired by the network through a learning process.
2. Interneuron connection strengths known as synaptic weights are used to store the knowledge. [3]

An Artificial Neural Network is a network of many simple processors ("units"). The units are connected by communication channels ("connections") which usually carry numeric (as opposed to symbolic) data, encoded by any of various means. The units operate only on their local data and on the inputs they receive via the connections. The restriction to local operations is often relaxed during training. The weighted sum of inputs to unit j is given by

$$net = \sum_i w_{ji}\, i_{i.} \qquad w_{ji} \text{ is the weight connecting the } i^{th} \text{ input to the } j^{th} \text{ neuron} \ldots\ldots\ldots\ldots(1)$$

The design of a neural network begins with network architecture. The task of navigation and obstacle avoidance together is treated as a function approximation problem. Hence, a feed-forward architecture is used to model the problem. The weighted sum of all the inputs is fed to each layer of the network in succession. The output of one layer becomes the input of the next until the next layer is reached.

Another important design aspect is the choice of the training algorithm.

Back-Propagated Delta Rule Networks (BP)

Backpropagation is a development from the simple Delta rule in which extra hidden layers (layers additional to the input and output layers, not connected externally) are added. The network topology is constrained to be feedforward: i.e. loop-free. Generally connections are allowed from the input layer to the first (and possibly only) hidden layer; and so on to the output layer. The hidden layer learns to recode (or to provide a representation for) the inputs. More than one hidden layer can be used. The architecture is more powerful than single-layer networks; it can be shown that any mapping can be learned, given one hidden layer (of units).

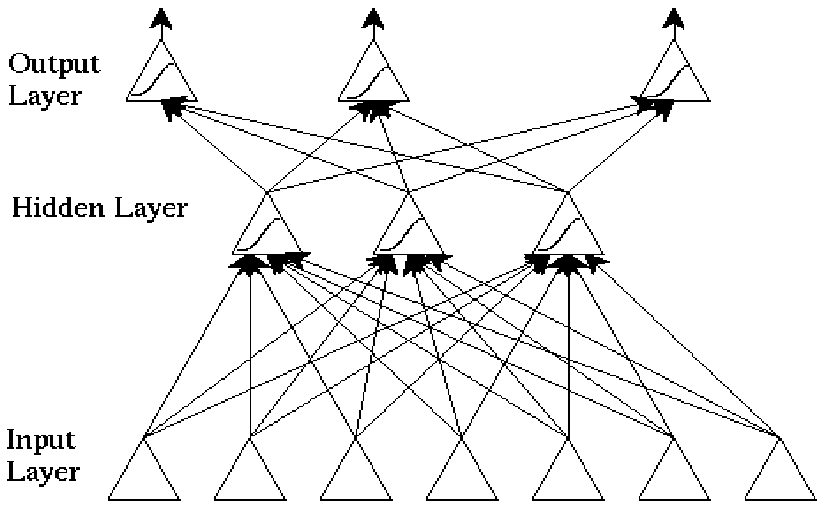


Fig.4 Feed forward Network Architecture.

3.1 Training BP Networks

The weight change rule is an application of the steepest descent search algorithm. Weights are changed by an amount proportional to the error at that unit times the output of the unit feeding into the weight.

Running the network consists of:

Forward pass: The outputs are calculated and the error at the output units calculated as the difference between the computed output and the desired output.

Backward pass: The output unit error is used to alter weights on the output units. Then the error at the hidden nodes is calculated (by back-propagating the error at the output units through the weights), and the weights on the hidden nodes altered using these values.

For each data pair to be learned, a forward pass and backwards pass is performed. This is repeated for a fixed number of iterations or until a predetermined acceptable error is reached.

The neuron or processing unit:

Analog processing units with sigmoidal activation function have been used.

As a function:

$$f(net) = 1/(1 + \exp(-k*net)) \quad \ldots\ldots (2)$$

Fig 5. shows the output for k=0.5, 1, and 10, as the activation varies from -10 to 10.

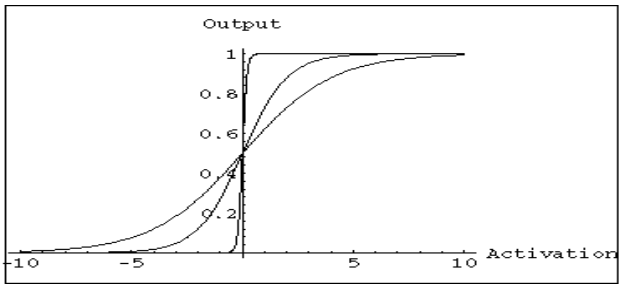


Fig.5 Activation Function (sigmoid)

3.2 Input Representation

The most important factor determining the performance of a neural network on a given task is the input representation. For autonomous navigation and obstacle avoidance of a mobile robot, as described earlier, we use one CCD camera to capture the images of the obstacle course. The input images are analog, i.e. continuous color, images. Before providing these images to the network, they are converted into binary form. The justification of this step lies in the fact that outdoor lighting conditions may vary. Also the obstacles and the guidelines (the 2 lines which are 10 ft. apart) may be white or yellow. Converting to binary form simplifies the task of the network at a very little computation cost. This is followed by resizing the image to a frame of 30 x 30 pixels. The track conditions are such that most of the information contained in the image is redundant. Also resizing, which can be easily achieved by a standard algorithm, reduces the size of the network drastically.
The binary values of the pixel i.e. 0 or 1 are used as input to the network. The 30 x 30-image matrix is vectorized to a 900 element vector when it is fed to the network. The preprocessing performed on the image seeks to reduce the complexity of the problem by standardizing the "features" needed for navigation and avoiding the obstacles. The task of identifying these features is left to the network.

3.3 Output Representation

The next crucial decision in determining the networks' mechanism is the output representation. The one out of N- coding technique was used to represent the desired steering response to a given condition. The steering angle is continuous and bounded in an interval. This interval is descretized to finite number (5 in this case) of responses.

Steering Angle in degrees	Output Vector
-10	[1 0 0 0 0]
-20	[0 1 0 0 0]
0	[0 0 1 0 0]
10	[0 0 0 1 0]
20	[0 0 0 0 1]

Table 2.Output Representation.

The parameters of the network, called synapses or weights, have to be fixed before the network is put to use. Training performs this task by iterative updating of randomly initialized weights. The back propagation algorithm was used for this purpose. Since this problem does not require knowledge of previous states, recurrent models were not considered. Of the other available algorithms, the unsupervised learning algorithms were considered unsuitable since they are incapable of providing particular desired responses [4]. Also, since training is off-line, the speed of convergence is of less consequence. Another advantage of training off-line is that structured noise can be added to increase the network's ability to handle a variety of situations.

4.Comparison of the Rule Based Model vs. the Connectionist Model

Before delving into a comparison of the results from applying the different paradigms, it is interesting to note the basic differences between them.

The Rule based algorithm relies on specifying the input variables explicitly; in other words, using exclusively the features identified as being influential in determining the output drives the output. It is critical to identify all the important features and only these features. Missing a critical feature will lead to unpredictable results, while having extraneous features increases the size of the rule base, making the algorithm inefficient. Therefore, it is said that a fuzzy system is modeled by drawing inferences from the real system.

The Neural Network is trained using real instances of the system. This is called learning from examples. The purpose of training is to fix the weights of the network in such a way that the influential features are highlighted. This method ensures that all the critical features are identified. However, the accuracy of the network depends on the training data provided to it. It is quite possible that ill-chosen examples bias the network output. Also the network may pick up features present in examples by coincidence, since it has no way of distinguishing relevant features from irrelevant ones.

In addition to the explicit inputs, it is necessary to come up with a set of rules to be able to model a fuzzy system. Fuzzy systems are amenable for use only in situations where the inputs and outputs have explicit relations. Once these relations are identified, coding a fuzzy system is very straightforward. The parameters and fuzzy sets are defined according to experience, making the system transparent, hence, easy to maintain and upgrade. A typical program consists of a set of IF-THEN statements. More often than not some form of preprocessing is required for numerical input and output representation. In its simplest form, this could be scaling. As discussed earlier, a neural network relies on training to relate inputs and outputs. Although observing the weights gives us an idea about the input –output relations, the governing rules cannot be explicitly stated. Thus training a neural network is a black box.

The mechanism of a fuzzy logic program involves running through a series of conditional (If-then) statements. In the worst case, all the rules have to be checked for, which can lead to high computational time. A neural network involves simple (multiplication) weighted summations equivalent to matrix multiplication. A feed-forward neural network has a fixed finite number of such computations on every operation.

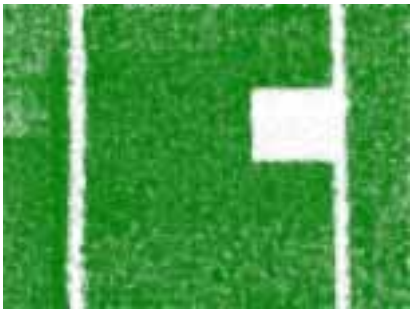
As the number of input variables or resolution into linguistic variables of a fuzzy system increases, the rule base increases in size geometrically. A feed forward neural network with a single hidden layer can model highly non-linear systems. This is due to the massively parallel nature of the network. It must be mentioned, however, that no deterministic methods exist to determine the optimum number of hidden layer of neurons required to model a given system. Also, any change in size of the network means that the network has to be trained again.

Any errors during the formulation of the rule base or while coding severely affect the performance of the system, as there is a one to one correspondence between an output and a rule. Fuzzy systems are also sensitive to errors in data.
Neural Networks are relatively more robust and tolerant of noisy data. Since each neuron contributes to every neuron in the next layer failure of just a few does not greatly affect the network performance. The subtle relationship between the input and the output makes the network robust and capable of generalization.

5. Results: A) *Fuzzy Logic Controller:* Table 3.

Remarks:	**Distance Error:(de)**	**Angle Error: (ae)**	**Steering angle: (st)**
Case 1:Ideal	0	0	0
Case 2:	2	0	-5.2
Case 3:Extreme	3	0	-10
Case 4:Extreme	-3	0	10
Case 5:Extreme	0	20	19.6
Case 6:Extreme	2.5	20	-19.6
Case 7:Extreme	-3	20	19.6
Case 8:	-1.5	-1	10
Case 9:	1.5	10	-13.6
Case 10:	-2	13	10.8

B) *Neural Networks:* The following test images were presented to the network. The output steering direction is shown below.



Test Image 1: Small Left Turn 10 deg

Test Image 2: Small Right Turn 10 deg

Test Image 3: Sharp Left Turn 20 deg

Fig 6. Test Images

6.Conclusions and Future Work:

The Summary Table is as follows:

Criteria for Comparison	Fuzzy Controller	Neural Networks
Principle	Thumb rules for navigation of the robot in the obstacle course inferred from human drivers actions.	Human drivers' actions mimicked for "learning" by the network.
Inputs and preprocessing required	Calibration of camera required to obtain the d_{error} and θ_{error} –the input variables	Raw image data converted to binary form and resized to 30 x30 frame
Complexity of Algorithm	In the worst case 32 conditionals (If-Then statements) to be evaluated.	For a network of size 'n' the same number of multiplication and addition operations are required.
Size of the Model	Increases geometrically with increase in resolution or number of input variables.	Increases linearly with increase in number if processing units.
Robustness of the algorithm	Sensitive to noise in data and coding errors.	Massive parallel structure gives robustness and ability to generalize.
Coding and Maintenance	Transparent code helps in maintenance.	Involves matrix operations for efficient implementation.
Hardware Implementation	Relatively difficult.	Easily hard wired.
Tuning	No analytical technique exists, based purely on trial and error.	Tuning can be achieved by suitable training algorithms
Stability	Difficult to develop a model that can prove the stability of the controller.	A feed forward network with one hidden layer can provably model highly non-linear systems.

Table 4.Comparison between the two techniques

As discussed earlier, the tuning of the fuzzy logic controller is still an open problem. No analytical techniques are known to exist for the tuning of the rule-based system. Hence it is difficult and time consuming to come up with a set of thumb rules for the rule-based system. On the other hand, the neural network behaves like a black box. The relationship between inputs and outputs of the network is subtle and cannot be extracted explicitly. [5]

Incorporating speed control in the fuzzy system entails adding another layer of rules above the existing rule-base [2]. The information required for speed control is the desired steering angle and the position of the obstacle with respect to the robot. Such a two-layered system increases the computational cost tremendously. The raw image data presented to the neural network does not contain information necessary to achieve speed control. Implementing speed control requires development of a different input and output representation.

The general and problem specific limitations of the two systems are being addressed by a new field of research called fuzzy-neural systems or neuro-fuzzy systems. Tuning of fuzzy systems can be achieved by neural networks. In a different model,

fuzzy neurons, with membership functions as activation functions are used to model fuzzy-logic controller with learning ability [6]. The number of input neurons is equal to the number of rules. Other hybrid models that utilize the similarities of fuzzy and neural systems to develop more versatile solutions provide directions for future work.

7. References:

1. Tayib Samu, Nikhil Kelkar, David Perdue, Mike Ruthemeyer, Bradley Matthews and Ernest Hall, "Line following using a two camera guidance system for a mobile robot," SPIE Conference 2904-39, Boston, MA, November 1996.
2. Nikhil Kelkar and E.L. Hall, "Fuzzy Logic Control of an AGV," Proc. of Intelligent Robots and Computer Vision XVI, Oct. 15-17, 1997, Pittsburgh, PA.
3. Simon Haykin," **Neural networks a comprehensive foundation**", Maxwell Macmillan International, New York, 1994.
4. Dean A Pomerleau, "**Neural Network Perception for Mobile Robot Guidance**", Kluwer Academic Publishers, Boston, MA, 1993.
5. Toru Yamaguchi, Kenji Goto and Tomohiro Takagi, "**Applied Research in Fuzzy Technology**", Kluwer Academic Publishers, Boston, MA, 1994.
6. Robert J. Schalkoff, "**Artificial Neural Networks**", The McGraw Hill Companies Inc., New York, 1997.