بسم الله الرحمن الرحيم

SUDAN UNIVERSITY OF SCIENCE AND TECHNOLOGY

COLLEGE OF GRADUATE STUDIES

Nero-Fuzzy Controller Design for Permanent Magnet DC Motor

تصميم متحكم عصبي غامض لمحرك تيار مستمر ذو المغنطيس الدائم

A Thesis Submitted in Partial Fulfillment for the Requirements of the Degree of M.Sc. in Electrical Engineering (Microprocessor and Control)

Prepared by:

Elsiddig Elsheikh Eltayeb

Supervised by:

Dr. Awadalla Taifour Ali

May 2014

آية

بسم الله الرحمن الرحيم

إنَّ {في خلْقِ السَّماواتِ والأرضِ واختلافِ اللَّيلِ والنَّهارِ لآياتٍ لأولي الألباب *الَّذين

يذْكُرونَ الله قياماً وقعوداً وعلى جُنوبهم ويتفكَّرونَ في خلْقِ السَّمواتِ والأرضِ ربَّنا ما

خلَقْتَ هذا باطلاً سبحانَك فَقِنَا عذابَ النَّار}

(آل عمران آية 191.190)

DEDICATION

Dedicated specially to

my beloved mother, father, wife, son, brothers, sisters, friends and teachers

for your care, support and believe in me

with my respect

ACKNOWLEDGMENTS

In the name of Allah, Most Gracious, and Most Merciful Praise be to Almighty Allah who gave us the courage and patience to carry out this thesis. My deep appreciation and heartfelt gratitude goes to my supervisor, **Dr. AWADALLA TAYFOR ALI** for his kindness, constant endeavor, guidance and the numerous moments of attention he devoted throughout this thesis. Our sincerely appreciate the efforts that have been put by all the people who guided me to the true way, which truly cannot be describe by words. Thank you very much to all for improving the quality of the thesis.

I would like also to thank the staff of Electrical engineering department, college of engineering, who helping me in different ways. Also I would like to acknowledge **Mr. Omer Salama** and **Mr. Glal A.rahman** for their Assistance, support and encouragement.

ABSTRACT

This thesis presents a neuro-fuzzy controller design for speed control of Direct Current (DC) motor. The thesis scope includes the simulations and modeling of DC motor, Fuzzy Logic Controller (FLC), neuro fuzzy controller and conventional Proportional-Integral-Derivative (PID) controller as benchmark to the performance of fuzzy system. The most commonly used controller for the speed control of DC motor is the conventional PID controller. Fuzzy logic controller and neuro-fuzzy control are proposed in this study. The performances of the two controllers are compared with PID controller performance. Classical control theory is based on the mathematical models that describe the physical plant under consideration.

In this thesis, neural networks are used in to solve the problem of tuning a fuzzy logic controller. The neuro fuzzy controller uses neural network learning techniques to tune membership functions. Comparison between the PID output, FLC output and the neuro fuzzy output was done on the basis of the simulation result obtained by MATLAB/SIMULINK. The model Performance of neuro-fuzzy controller is better compared to FLC and PID controller.

مستخلص

تقدم هذه الدراسة تصميم متحكم عصبي غامض للتحكم في سرعة محرك تيار مستمر. محتوي الدراسة يشمل النموذج الرياضي لمحرك تيار مستمر وعمل محاكاة لأداء المحرك. تم تطبيق نوعين من انواع التحكم وهي المتحكم الغامض ومتحكم تقليدي يعرف بالمتحكم التناسبي التكاملي التفاضلي. استخدمت المتحكمات كمقياس لأداء المحرك. ويعتبر المتحكم التناسبي التكاملي التفاضلي الاكثر استخداماً للتحكم في سرعة محركات التيار المستمر. الفكرة الاساسية لهذه الدراسة هي مقارنة المتحكم العصبي الغامض مع المتحكم التناسبي التكاملي التفاضلي و المتحكم الغامض. في هذه الدراسة تم استخدام تقنيات تعلم الشبكات العصبية كطريقة جديدة لحل مشكلة دالة العضوية في التحكم الغامض. تم عمل محاكاة لأداء النظام مع انواع المتحكمات الثلاث باستخدام برنامج MATLAB/SIMULINK. ووجد أن أداء المتحكم العصبي الغامض أفضل من أداء المتحكم التقليدي والمتحكم الغامض.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AC	Alternative Current
ANFIS	Adaptive Neuro Fuzzy Inference System
ANNs	Artificial Neural Network
BP	BackProbagation
BOA	Bisector Of Area
C-C	Cohen-Coon
DC	Direct Current
FC	Fuzzy Controller
FLC	Fuzzy Logic Controller
FIS	Fuzzy Inference System
GUI	Graphical User Interface
IAE	Integral of Absolute Error
IMC	Internal-Model-Control
ISTE	Integral of Squared Time weighted Error
MATLAB	Matrix Laboratory
MFs	Membership Functions
MLP	Multi Layered Perceptron
NARMA	Non-linear Auto-Regressive Moving Average
NNC	Neural Network Controller
NT	No Definite Trend, Minor Change
PID	Proportional Integral Derivative
PM	Permanent Magnet
Z-N	Ziegler - Nichols

LIST OF SYMBOLS

ce	change of error
e	error
K_D	Derivative gain
K_I	Integral gain
K_P	Proportional gain
r(t)	Reference Input signal
u(t)	Control signal
ω(t)	Angular velocity
y(t)	Plant Output Signal
μ(x)	Degree of Membership Function

CHAPTER ONE

INTRODUCTION

CHAPTER ONE

INTRODUCTION

1.1 General Overview

Accurate control is critical to every process that leads to various types of controllers which are being widely used in process industries. Tuning methods for these controllers are very important for process industries. DC motor has been selected because it is widely used in industrial applications, robot manipulators and home appliances where speed and position control are required. The DC motors can come in many shapes and sizes, makes the development of DC motor application quite easy and flexible. It is also has high reliabilities, flexibilities and low cost [1]. To overcome these difficulties, there are three basic approaches to intelligent control: knowledge based expert systems, fuzzy logic, and neural networks. All three approaches are interesting and very promising areas of research and development [2].

Classic control has proven for a long time to be good enough to handle control tasks on system control; however its implementation relies on an exact mathematical model of the plan to be controlled and not simple mathematical operations. Intelligent, self-learning or self-organizing controls using expert systems, artificial intelligence, fuzzy logic, neural networks, hybrid networks,… etc have been recently recognized as the important tools to improve the performance of the systems in the industrial sectors. Combination of this intelligent controls with the adaptiveness appears today as the most promising research area in the practical implementation and control of electrical drives.

1.2 Problem Statement

All control systems suffer from problems related to undesirable overshoot, longer settling times and vibrations and stability while going from one state to another state. Real world systems are nonlinear, accurate modeling is difficult, costly and even impossible in most cases conventional PID controllers generally do not work well for non-linear systems. Therefore, more advanced control techniques need to be used which will minimize the noise effects.

Design of a fuzzy logic controller is accompanied with certain problems regarding design of membership functions (type and number of membership functions, their shape and range, etc.), and choosing appropriate fuzzy rules. Moreover, developing a rule base is one of the most time-consuming parts of designing a fuzzy logic controller. Usually it is very difficult to transform human knowledge and experience into a rule base of fuzzy logic controller. Frequently, designing a fuzzy logic controller requires a number of trial and error iterations, and even then, it is very difficult to ensure that the designed controller is an optimal one. Hence there is a need for developing efficient methods to tune membership functions, i.e. to obtain optimal shapes, ranges and number of membership functions, etc. and to obtain optimal rule base. Neural networks do not produce an explicit model even though new cases can be fed into them and new results obtained. Neural nets may not provide the most cost effective solution neural net implementation is typically more costly than other technologies, in particular fuzzy logic. The combination of neural networks and fuzzy logic controller has received attention. The idea is to lose the disadvantages of the two and give the advantages of both. NNs bring into this union a model of the system based on membership functions and rule base.

1.3 Objectives

The main aims of this study are:

- To optimize fuzzy logic controller parameters via neural network and use the neuro-fuzzy scheme to design controller for a Permanent Magnet Direct Current (PMDC) motor.
- To present special neural network architecture that can be converted to fuzzy logic controller.
- To design a neuro-fuzzy controller able to precisely learn the control relation between input-output training data generated by the learning algorithm.
- To perform a design simulation of neuro-fuzzy logic controller done by using MATLAB/SIMULINK.
- To simulate speed control of DC motor using conventional PID controller as a comparison for fuzzy logic controller and neuro-fuzzy controller in the same range.

1.4 Methodology

In this study a Back Propagation (BP) algorithm is used to learn the neural network. The BP algorithm learns the weights for a multilayer network, given a network with a fixed set of units and interconnections. The type of fuzzy inference systems that can be implemented in the fuzzy logic toolbox: Mamdani-types. This type of inference systems vary somewhat in the way output are determined. The mamdani-style fuzzy inference process is performed in four steps:

- Fuzzification of the input variable.
- Rule evaluation (inference).
- Aggregation of the rule outputs (composition).
- Defuzzification.

The task is to design and display the simulation of the neuro-fuzzy controller and the result of the simulation will be display by using rule viewer and surface viewer which are parts of the Graphical User Interface (GUI) tools in fuzzy logic toolbox and neural network toolbox packages in MATLAB TOOLBOX programming.

1.5 Thesis Outline

The thesis organisms consist of five chapters:

Chapter one presents a general overview, problem statement, thesis objective and Methodology.

Chapter two contains all the literature review from the previous study related to the objectives of the study. Chapter two also concerns with general introduction of the main ideas of DC motor control and general ideas, modeling, PID controller, fuzzy logic controller, neural network controller and neuro-fuzzy controller.

Chapter three presents the overall design, mathematical model, SIMULINK model and methods to apply in the study. The main methodology that been stressed out related speed control of PMDC motor using PID controller , fuzzy logic controller and neuro-fuzzy controller, also has been explained in chapter three.

Chapter four consists of expected outcomes that been stated based on the objectives of the study. It shows that by using neuro-fuzzy controller the speed controller of DC motor can be simulated and the discussion that achieved in the study.

The last chapter handles the conclusion and recommendations for future studies.

CHAPTER TWO

THEORETICAL BACKGTOUND AND LITERATURE REVIEW

CHAPTER TWO

THEORETICAL BACKGTOUND AND LITERATURE REVIEW

2.1 Introduction

This chapter highlights the literature cited on the DC motors, PID controller, fuzzy logic controller, neural network controller and neuro fuzzy controller. Because of their high reliabilities, flexibilities and low costs, DC motors are widely used in industrial applications, electric trains, robot manipulators and home appliances where speed and position control of motor are required. PID controllers are commonly used for motor control applications because of their simple structures and intuitionally comprehensible control algorithms. Fuzzy set theory (Zadeh, 1965) which led to a new control method called fuzzy control which is able to cope with system uncertainties.

The mathematical model of a DC motor is used to obtain a transfer function between rotor shaft speed and applied armature voltage. However, there is no systematic method for designing and tuning the fuzzy logic controller and one has to design using some trial and error using the IF, ELSE, THEN rules. Furthermore, an optimal fuzzy logic controller cannot be achieved by trial-and-error. These drawbacks have limited the application of fuzzy logic control. Some efforts have been made to solve these problems and simplify the task of tuning parameters and developing rules for the controller [3-4]. These approaches mainly use adaptation or learning techniques drawn from artificial intelligence or neural network theories. This logic combined with neural networks yields very significant results. The learning and identification of fuzzy logic systems need to adopt techniques from other areas, such as statistics, system identification. Since neural networks can learn, it is natural to merge these two techniques. This merged technique of the learning power of the neural networks with the knowledge representation of fuzzy logic has created a new hybrid technique, called as the term ‘neuro-fuzzy networks’[5]. This model is then built in MATLAB/SIMULINK. Then design and tuning of PID controllers, fuzzy logic controller and neuro-fuzzy controller are reviewed in SIMULINK with the proposed design procedure.

2.2 DC Motor

DC motors are widely used in many applications of daily life. They are found them everywhere, from house appliances to our vehicles, desktops and laptops, and industrial applications such as production lines, remote control airplanes, automatic navigation systems and many other applications [1]. DC motors are well known for their torque-speed characteristics, and their wide operation voltage and current range. DC motors can be specified into different types: permanent magnet motors, shunt motors, series motors and compound motors. For these DC motor types, each one of them has different speed-torque characteristics and different categories of motors. DC servomotors are permanent magnet motors, in which speed and position are typically the most common parameters to control. Basic DC motors as used on nearly all packaged drives have a very simple performance characteristic the shaft turns at a speed almost directly proportional to the voltage applied to the armature. However, when operated at a fixed applied voltage but a gradually increasing torque load, they exhibit a speed drop. In the DC drive a similar type of "compensation" is employed in the control to assist in maintaining a nearly constant speed under varying load (torque) conditions.

In this thesis, the PMDC motor model is chosen according to his good electrical and mechanical performances more than other DC motor models. The DC motor is driven by applied voltage. PMDC motor is found in a wide variety of low-power applications. The field winding is a permanent magnet. Permanent magnets offer a number of useful benefits such as:

1. They do not require external excitation.
2. Less space requirement.
3. They are cheaper.

Field controlled DC motor is open loop while armature controlled is closed loop system. Hence armature controlled DC motor are preferred over field controlled system for small size motor field control is advantageous because only a low power servo amplifier is required while the armature current which is not large can be supplied from an expensive constant current amplifier. For large size motor it is on the whole cheaper to use armature control scheme. Further in armature controlled motor, back emf contributes additional damping over and above that provided by load friction.

The speed of DC motor can be adjusted to a great extent as to provide controllability easy and high performance [6].The term speed control stand for intentional speed variation carried out manually or automatically DC motors are most suitable for wide range speed control and are there for many adjustable speed drives. DC motor there are basically three method of speed control:

1. Variation of resistance in armature circuit.
2. Variation of field flux.
3. Variation of armature terminal voltage.

The controllers of the speed that are conceived for goal to control the speed of DC motor to execute one variety of tasks, is of several conventional and numeric controller types, the controllers can be: PID controller, fuzzy logic controller; or the combination between them: fuzzy-neural networks, fuzzy-genetic algorithm, fuzzy- Ants colony, fuzzy-swarm. They are several methods of speed control:

i. Traditionally rheostat armature control method was used for low power DC motors.

ii. Use of conventional PID controllers.

iii. Neural Network Controllers (NNC).

iv. Constant power field weakening controller based on load-adaptive multi-input multi-output linearization technique (in high speed regimes).

v. A single phase uniform PWM Alternative Current (AC)-DC buck-boost converter with only one switching device used for armature voltage control.

vi. Use of NARMA-L2 (Non-linear Auto-Regressive Moving Average) controller in the constant torque region.

2.3 PID Controller

Elmer Sperry created the first PID type controller in 1912 to help with ship steering (Bennett, 1979). A PID controller is referred to as a three-term controller using a proportional term, integral term, and derivative term combined in a linear algorithm. The proportional term calculates the gain based on present error. The integral term calculates the sum of all past errors. The derivative term uses the rate at which the error has been changing to predict future error. This controller also uses a feedback loop to compensate for error. The error is described as the difference between the desired set point of the system and the measured variable calculated by the P, I, and D terms. Once a PID controller is designed, a tuning process must follow in order for the controller to meet the needs of a specific system.

When working with applications where control of the system output due to changes in the reference value or state is needed, implementation of a control algorithm may be necessary. The PID controller can be used to control any measurable variable, as long as this variable can be affected by manipulating some other process variables. Many control solutions have been used over the time, but the PID controller has become the 'industry standard' due to its simplicity and good performance [7]. However, these controllers provide better performance only at particular operating range and they need to be retuned if the operating range is changed. Further, the conventional controller performance is not up to the expected level for nonlinear and dead time processes. In the present industrial scenario, all the processes require automatic control with good performance over a wide operating range with simple design and implementation block diagram of the drive with PID controller is shown in Figure 2.1. The speed error e(t) between the reference speed u(t) and the actual speed w(t) of the motor is fed to the P-I-D controller. The transfer function of the most basic form of PID controller is given by:

$$C(s) = K_P + \frac{K_I}{s} + K_D s = \frac{K_D s^2 + K_P s + K_I}{s} \qquad (2.1)$$

Where K_P = Proportional gain, K_I = Integral gain and K_D = Derivative gain.

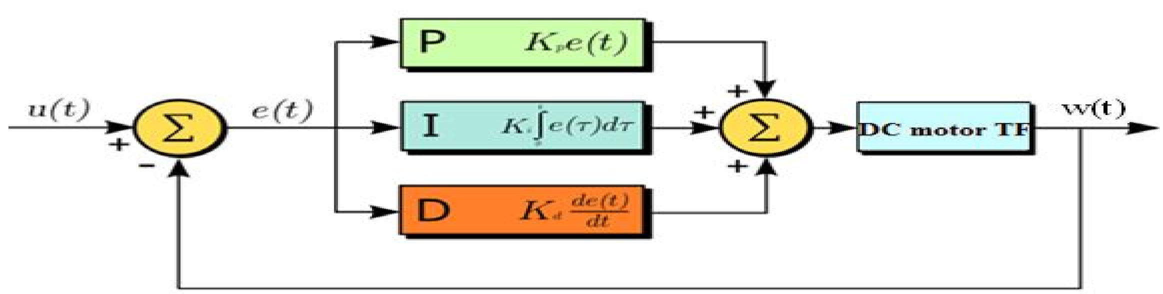

Figure 2.1: A schematic of a system with a PID controller

In PID controller structure we assume the controller is used in a closed-loop unity feedback system as shown in Figure 2.1. PID controller is used in more than 95% of closed-loop industrial processes. It can be tuned by operators without extensive background in controls, unlike many other modern controllers that are much more complex but often provide only marginal improvement. In fact, most PID controllers are tuned on-site. They are four major characteristics of the closed-loop step response that use to understand the PID parameters affect on system dynamics:

- Rise time: the time it takes for the plant output y to rise beyond 90% of the desired level for the first time.
- Overshoot: how much the peak level is higher than the steady state, normalized against the steady state.
- Settling time: the time it takes for the system to converge to its steady state.
- Steady-state error: the difference between the steady-state output and the desired output.

The effects of increasing each of the controller parameters P, I and D can be summarized in Table 2.1

Table 2.1: The PID parameters affect on system dynamics

Response	**Rise Time**	**Overshoot**	**Settling Time**	**S-S Error**
K_P	Decrease	Increase	Not change	Decrease
K_I	Decrease	Increase	Increase	Eliminate
K_D	Not change	Decrease	Decrease	Not change

Table 2.1 used for designing a PID controller by the following typical steps:

• Determine what characteristics of the system need to be improved.
• Use K_P to decrease the rise time.
• Use K_D to reduce the overshoot and settling time.
• Use K_I to eliminate the steady-state error.

Several approaches were developed for tuning PID controller such as the Ziegler-Nichols (Z-N) method, the Cohen-Coon (C-C) method, Integral of Squared Time weighted Error (ISTE) rule, Integral of Absolute Error (IAE) criteria, and Internal-Model-Control (IMC) based method and gain-phase margin method [8]. PID controllers are usually tuned using hand tuning or Ziegler-Nichols methods to obtain the desired performance according to preset criteria. The basic continuous feedback control is PID controller. The PID controller exhibits good performance but is not adaptive enough (Oyas and Nordin, 2008).

2.4 Artificial Neural Network

The science of Artificial Neural Networks (ANNs) is based on the biological neuron shown in Figure 2.2. In order to understand the structure of artificial networks, the basic elements of the biological neuron should be understood. Neurons are the fundamental elements in the central nervous system. A biological neuron, it essentially has four main parts: dendrites, cell body, axon, and synapses terminals. The dendrites are branching structures that receive electrical impulses or signals from other neurons. The dendrites send their signals to the body of the cell. The cell body contains the nucleus of the neuron and organelles, but functionally processes the incoming signal from the dendrites. If the sum of the received signals is greater than a threshold value (48mv), the neuron fires by sending an electrical pulse along the axon to the next neuron. The axon is the portion of the neuron that takes the electrical impulses or signals from the cell body to the pre-synaptic terminals. Pre-synaptic terminals form the end of the axon where it junctions with another neuron at a specialized location called a synapse. A synapse is where the axon of one neuron communicates with the dendrites of another neuron.

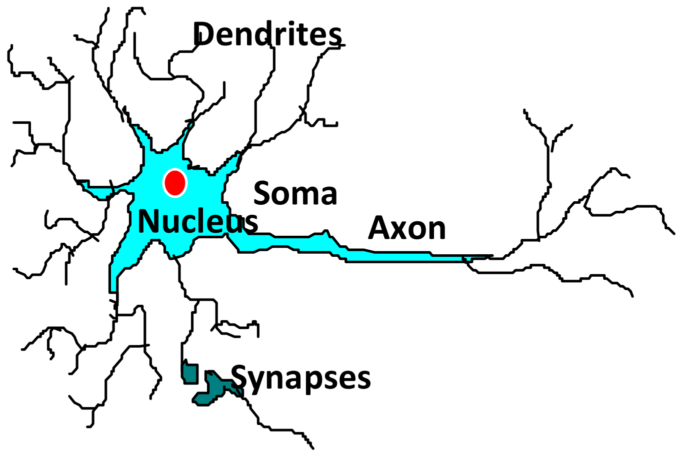

Figure 2.2: Basic elements of a neuron

Biological neurons are arranged in network architecture with vast numbers of neurons interconnected to each other allowing for rapid communication spanning throughout all areas of the body. Biological neural networks are much higher in complexity than this representation but it is basic structure that ANN's model.

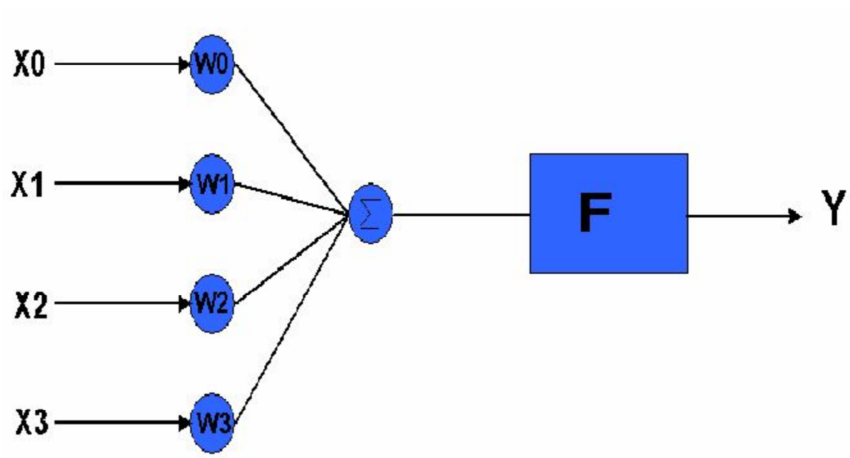

Figure 2.3: Neuron model

The model shown in Figure 2.3 is based on the components of the biological neuron The inputs X_0-X_3 represent the dendrites. Each input is multiplied by weights W_0-W_3. The output of the neuron model, Y is a function F of the summation of the input signals.

2.4.1 Literature review

At the beginning of 1800's scientists started to discover the nervous system in the human body, their work on knowing the structure and function of the nervous system continued until 1906 when they started to understand the basic operation of the neuron, and had a clear overview of how it operates and how the basic interconnection of neurons in the nervous systems looks like (J.J Hofield, 1982). After this great discover, McCulloch and Pitts in 1943 came up with the concept of artificial neural networks, In 1943 McCulloch and Pitts published a paper that discussed biological neuron function in the body, as well as going a step further to design and build a primitive artificial neural network made of simple electronics (McCulloch, & Pitts, 1943). ANNs are arranged in similar network architecture as their biological model; composed of singular and simplistic neurons that communicate rapidly through a network. ANNs have artificial neurons arranged in three basic layers. An ANNs starts with an input layer containing an equal number of neurons to inputs. A middle or hidden layer performs computations to create an output. The final layer, the output layer, sends the controller output to the plant portion of the system. Each artificial neuron, excluding input neurons in the first layer, can have multiple inputs. The artificial neuron sums the weighted inputs and formulates a single output that can be propagated to multiple neurons in the next layer after processing through an activation function. By combining multitudes of singular artificial neurons into a vast processing network, ANNs are capable of complex problem solving and control [9].

2.4.2 Advantages of ANNs

- The main advantage of neural networks is that it is possible to train a neural network to perform a particular function by adjusting the values of connections (weights) between elements. For example, if we wanted to train a neuron model to approximate a specific function, the weights that multiply each input signal will be updated until the output from the neuron is similar to the function.
- Neural networks are composed of elements operating in parallel. Parallel processing allows increased speed of calculation compared to slower sequential processing.
- ANN has memory. The memory in neural networks corresponds to the weights in the neurons.

2.4.3 Disadvantage of ANNs

The main disadvantage of ANN is they operate as black boxes. The rules of operation in neural networks are completely unknown. It is not possible to convert the neural structure into known model structures such as ARMAX,…etc. Another disadvantage is the amount of time taken to train networks. It can take considerable time to train an ANN for certain functions.

2.4.4 Types of learning

Neural networks have three main modes of operation supervised, reinforced and unsupervised learning [10].

- In supervised learning (i.e. learning with a teacher), the output from the neural network is compared with a set of targets, the error signal is used to update the weights in the neural network.
- Reinforced learning, (i.e. learning with limited feedback) is similar to supervised learning however there are no targets given, the algorithm is given a grade of the ANN performance.

- Unsupervised learning, i.e. learning with no help updates the weights based on the input data only. The ANN learns to cluster different input patterns into different classes.

2.4.5 Neural network structures

There are three main types of ANN structures single layer feedforward network, amulti-layer feedforward network and recurrent networks [10]. The most common type of single layer feedforward network is the perceptron. Other types of single layer networks are based on the perceptron model.

- **Single layer perceptron**

The details of the perceptron are shown in Figure 2.4.

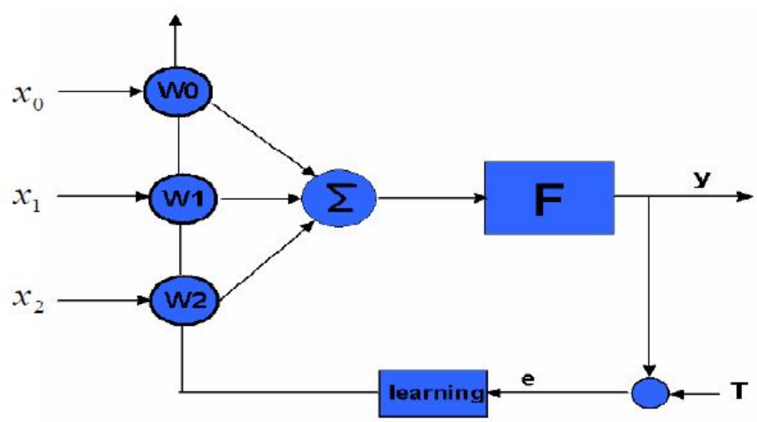

Figure 2.4: Perceptron model

Inputs to the perceptron are individually weighted and then summed. The perceptron computes the output as a function F of the sum. The activation function, F is needed to introduce nonlinearities into the network. This makes multi-layer networks powerful in representing nonlinear functions. There are three main types of activation function tan-sigmoid, log-sigmoid and linear shown in Figure 2.5 [9]. Different activation functions affect the performance of an ANN.

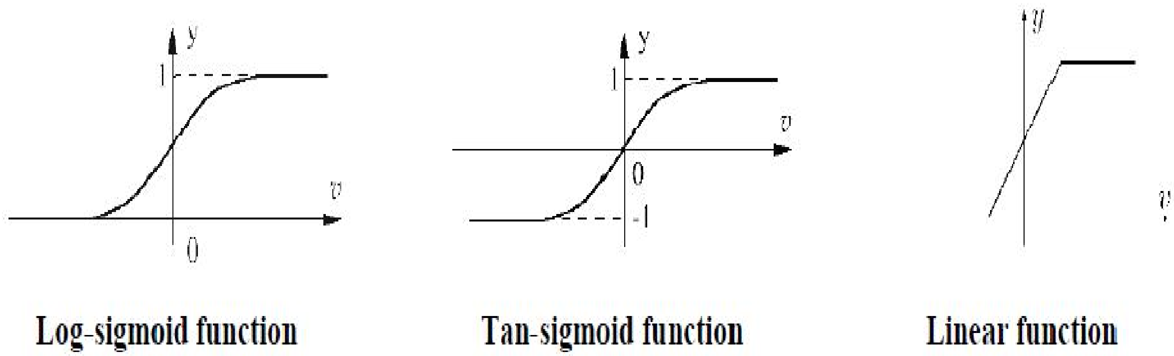

Figure 2.5: Types of activation functions

Single-layer feedforward networks are useful when the data to be trained is linearly separable. If the data trying to model is not linearly separable or the function has complex mappings, the simple perceptron will have trouble trying to model the function adequately.

- **Multi-layered perceptron**

Neural networks can have several layers. There are two main types of multi-layer networks feedforward and recurrent. In feedforward networks the direction of signals is from input to output, there is no feedback in the layers. This is shown in Figure 2.6.

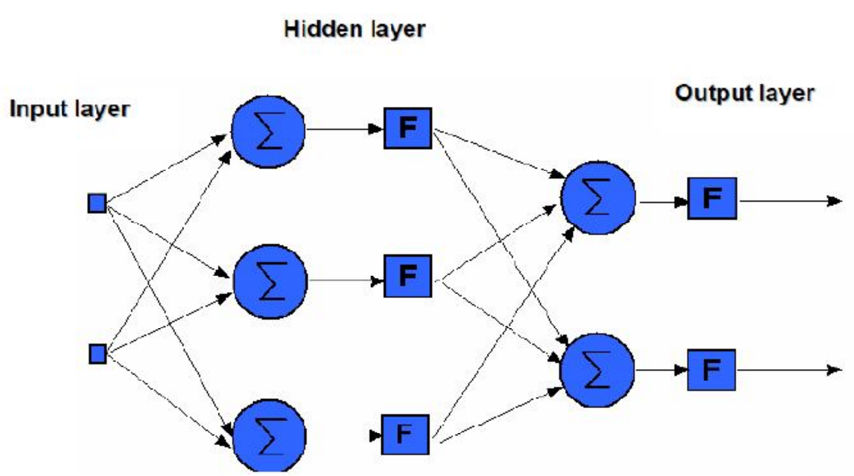

Figure 2.6: Multi layered perceptron

Increasing the number of neurons in the hidden layer or adding more hidden layers to the network allows the network to deal with more complex functions. Cybenko's theorem states that, a feedforward neural network with a sufficiently large number of hidden neurons with continuous and differentiable transfer functions can approximate any continuous function over a closed interval. The weights in Multi Layered Perceptrons (MLP's) are updated using the backpropagation learning [11]. There is two passes before the weights are updated. In the first pass (forward pass) the outputs of all neurons are calculated by multiplying the input vector by the weights. The error is calculated for each of the output layer neurons. In the backward pass, the error is passed back through the network layer by layer. The weights are adjusted according to the gradient decent rule, so that the actual output of the MLP moves closer to the desired output. A momentum term could be added which increases the learning rate with stability. The second type of multi-layer networks is shown in Figure 2.7 recurrent networks have at least one feedback loop. This means an output of a layer feeds back to any proceeding layer.

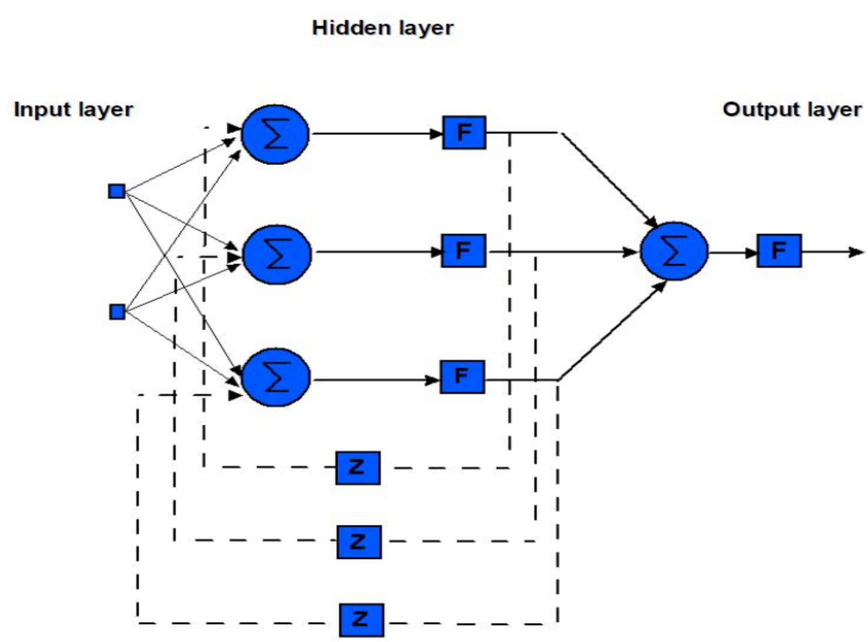

Figure 2.7: Recurrent neural network

This gives the network partial memory due to the fact that the hidden layer receives data at time t but also at time t-1. This makes recurrent networks powerful in approximating functions depending on time [11]. As an example the SIMULINK model for the nonlinear inverted pendulum shows that there are many feedback loops. This means the next state of the model depends on previous states. It is expected that to accurately model this type of dynamic system, a recurrent neural network with feedback loops will perform better than a static feedforward network.

- **Backpropagation**

Back-propagation neural network is a multilayer feed forward network with back-propagation of an error function (Xu et al., 2012). A simple back-propagation neural network has only three layers i.e. input, output and middle layer as shown in Figure 2.8.

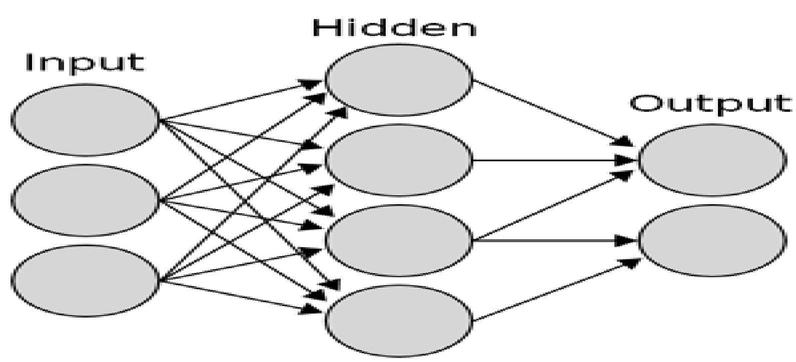

Figure 2.8: Layers of a feed forward neural network

The input weights are passed on to hidden layer for processing. The hidden layer passes calculated weights to the output layer. The error is presented to input layer through back propagation (feedback) when actual output is different from the desired level. Hence the weights are adjusted to minimize the error through training and learning of the neural network. The process continues until the output is acceptable or pre-configured learning time is achieved (Zhao et al., 2010). The Backpropagation is:

- Most common method of obtaining the many weights in the network.
- A form of supervised training.
- The basic backpropagation algorithm is based on minimizing the error of the network using the derivatives of the error function.

➢ Simple.

➢ Slow.

➢ Prone to local minima issues.

2.4.6 Applications of neural networks

In recent years, they have been widely used for optimum calculations and processes in industrial controls, communications, pattern recognition and classification, control systems, time-series prediction, fault diagnostic, medical applications, data analysis, compression and expansion chemistry and petroleum. They are on the rise for use in many highly sensitive control mechanisms such as flight controls and implementation of high security device

2.5 Fuzzy Logic

Fuzzy Logic (FL) is classified in the artificial intelligence category. “Fuzzy Sets” were introduced in 1965 by Lotfi Zadeh from the University of California at Berkeley (Zadeh, 1965). As an extension of the classical control theory. According to him classical control theory put too much emphasis on precision and therefore could not the complex system [12]. FL controllers can interpret data that falls in the gray area much like a human mind can make cognitive decisions when there is no distinct answer. Fuzzy logic is unlike many traditional logic systems in that the reasoning is approximate and not exact. It is this logic approximation also done by humans with commonsense reasoning that makes FL a form of artificial intelligence. Zadeh formulated a mathematical analysis allowing data partial membership of a set instead of distinct membership versus non-membership categories. Fuzzy sets allow for gradual transition of data classification with permissible overlap between membership groups. This revolutionary logic system provides a way to describe systems or data that may be too complex or ill-defined for traditional analysis using precise mathematical methods. Zadeh ideas were not presented as a method of control, but were later applied to control theory and fuzzy logic controllers evolved. Fuzzy logic is represented by three parts:

i. Linguistic variables in place of numerical values using natural language terms such as “very”, “not” or “most”.
ii. Fuzzy conditional statements to form IF, THEN statements.
iii. Fuzzy algorithms that creates an order to the rules or instructions (Zadeh,1990).

2.5.1 Fuzzy logic controller

Fuzzy Logic Control (FLC) is one of the most successful applications of fuzzy set theory, introduced by L.A Zadeh in 1965 and applied (Mamdani 1974) in an attempt to control system that are structurally difficult to model. Since then, FLC has been an extremely active and fruitful research area with many industrial applications reported [12]. In the last three decades, FLC has evolved as an alternative or complementary to the conventional control strategies in various engineering areas. Fuzzy control theory usually provides non-linear controllers that are capable of performing different complex non-linear control action, even for uncertain nonlinear systems. Unlike conventional control, designing a FLC does not require precise knowledge of the system model such as the poles and zeroes of the system transfer functions. Imitating the way of human learning, the tracking error and the rate change of the error are two crucial inputs for the design of such a fuzzy control system [13-14].

Fuzzy control is a control method based on fuzzy logic. Just as fuzzy logic can be described simply as "computing with words rather than numbers", fuzzy can be described simply as "computing with sentences rather than equations". A fuzzy controller can include the empirical rules, and that is especially useful in operating controlled plants [15]. Fuzzy control provides a formal methodology for representing, manipulating and implementing a human's heuristic knowledge of how to control a system. Fuzzy controllers are used in various control schemes. The most obvious one in the direct control where the fuzzy controller is in the forward path in a feedback control. Fuzzy controller block diagram is given in Figure 2.9, where we show a fuzzy controller embedded in a closed-loop control system. The plant outputs are denoted by y(t) , its inputs are denoted by u(t), and the reference input to the fuzzy controller is denoted by r(t). A fuzzy controller is used. It replaces the conventional controller, say, a PID controller.

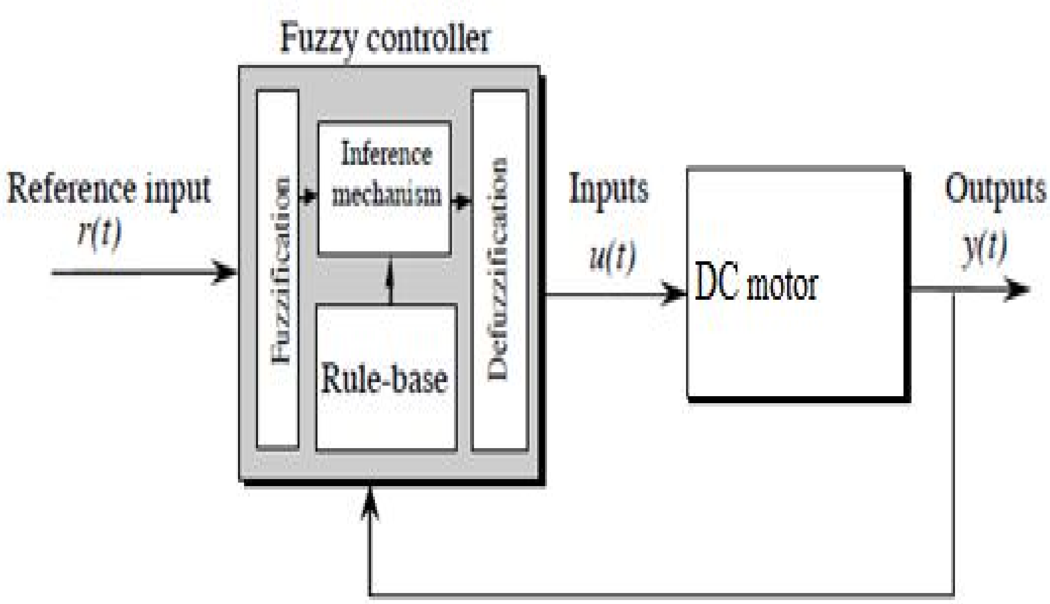

Figure 2.9: Fuzzy controller architecture

2.5.2 Structure of fuzzy controller

There are specific components characteristic of a fuzzy controller to support a design procedure. In the block diagram in Figure 2.10, the controller is between preprocessing block and a post-processing block. The following explains the diagram block by block.

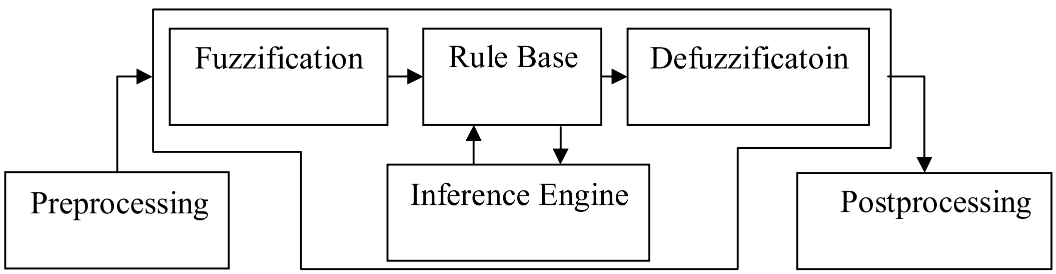

Figure 2.10: Fuzzy controller

(a) Pre-processing

The inputs are most often measured in hard or crisp from some measuring equipment, rather than linguistic. A preprocessor, the first block in Figure 2.14, conditions the measurements before they enter the controller. For this plant, the design preprocessing involves:

- Quantization in connection with sampling to integer.
- Normalization or scaling onto particular and standard rang.

(b) Fuzzification

Refer Figure 2.11 the first block in the controller is the fuzzification. It converts each input data to a certain degree of membership by a lookup in one or several membership functions. The fuzzification block matches the input data with the conditions of the rules to determine how well the condition of each rule matches that particular input instance. There is a degree of membership for each linguistic term that applies to that input variable.

- **Degree of membership (μ)**

The degree of membership (μ) is the degree to which a crisp variable belongs to a fuzzy set. It is expressed either as a fractional value ranging from 0.0 to 1.0 or percentage ranging from 0% to 100%.

- **Universe of discourse**

Elements of a fuzzy set are taken from a universe of discourse or just universe. The universe contains all elements that can come into consideration. Before designing the membership function it is necessary to consider the universe for the inputs and outputs. Naturally, the membership function for N and P must be defined for all possible values of error and change in error, and a standard universe may be convenient.

For this development this plant, three membership functions is considered, there are error (e), change of error (ce) and change of control signal (u) which the range (universe of discourse). The universe of discourse is considered based on environment and equipment condition and also considers other possibility might be occurred and influences the performance of system.

- **Membership function**

Every element in the universe of discourse is a member of a fuzzy set to some grade, maybe even zero. The grade of membership for all its members describes a fuzzy set, such as Z, P and N are Zero, Positive and Negative respectively. In fuzzy sets elements are assigned a grade of membership, such that the transition from membership to non-membership is gradual rather than abrupt. The set of elements that have a non-zero membership is called the support of the fuzzy set. The function that ties a number to each element x of the universe is called the Membership Functions (MFs).

• Start with triangular sets. All membership functions for a particular input or output should be symmetrical triangles of the same width. The leftmost should be shouldered ramps.

• The overlap should be at least 50%.The width should initially be chosen so that each value of the universe is a member of at least two sets, except possibly for elements at the extreme ends. If, on the other hand, there is a gap between two sets no rules fire for values in the gap. Consequently the controller function is not defined.

Membership function can be flat on the top, piece-wise linear and triangle shaped, rectangular, or ramps with horizontal shoulders.

➢ **Determine the MFs**

❖ Use the knowledge of human experts

❖ Data collected from various sensors

In order to define fuzzy membership function, designers choose many different shapes based on their preference and experience. There are generally four types of membership functions used:

✓ Trapezoidal MF.

✓ Triangular MF.

✓ Gaussian MF.

✓ Generalized bell MF.

Among them the most popular shapes are triangular and trapezoidal because these shapes are easy to represent designer's idea and require low computation time.

(c) Rule base

Fuzzy systems are knowledge based or rule based systems. The heart of a fuzzy system is a knowledge base consisting of the so- called If-Then rules. A fuzzy If-Then statement in which some words are characterized by continuous membership functions. After defining the fuzzy sets and assigning their membership functions, rules must be written to describe the action to be taken for each combination of control variables. These rules will relate the input variables to the output variable using If-Then statements which allow decisions to be made. The If (condition) is an antecedent to the Then (conclusion) of each rule, the controller may requires both the error, the change in error and the accumulated error as inputs, but we well call it a signal-loop control, because in principle all the three are formed from the error measurement. To simplify, the objective of this controller is to regulate some process output around a prescribed set point or reference [16].

➢ **Rule format**

Basically a linguistic controller contains rules in the if-then format, but it can be represented in different formats especially for complex system. Each rule in general can be represented in the following manner: If (antecedent) Then (consequence).

(d) Inference engine

Inference engine is defined as the software code which processes the rules, cases, objects or other type of knowledge and expertise based on the facts of a given situation. When there is a problem to be solved that involves logic rather than fencing skills, we take a series of inference steps that may include deduction, association, recognition, and decision making. An inference engine is an information processing system (such as a computer program) that systematically employs inference steps similar to that of a human brain. The inference mechanism provides the mechanism for invoking or inferring to the rule base such that the appropriate rules are fired. There are several inference procedures that can be used in FLCs listed as follow:

i. Max-Min.

ii. Max-Algebraic Product (or Max-Dot).

iii. Max-Bounded Product.

iv. Max-Drastic Product.

v. Max-Bounded Sum.

vi. Max-Algebraic Sum.

Two most common methods used in FLC are the max-min composition (introduced by Mamdani) and the max-algebraic product composition. The inference or firing with this fuzzy relation is performed via the operation between the fuzzified crisp input and the fuzzy relation representing the meaning of the overall set of rules. As a result of composition, one obtains the fuzzy set describing the fuzzy value of the overall control output. After inference process, the values is obtained by Δu will be used at defuzzification process and that values will be converted to a number that can be sent to the process as a control signal.

(e) Defuzzification

The resulting fuzzy set must be converted to a number that can be sent to the process as a control signal. This operation is called defuzzification and the resulting fuzzy set is thus defuzzified into a crisp control signal. There are several defuzzification methods is typical used and shown as follow [16].

- ✓ Centre Of Gravity (COG).
- ✓ Centre Of Gravity Method for Singletons.
- ✓ Mean Of Maxima (MOM).
- ✓ Bisector Of Area (BOA).
- ✓ Left Most Maximum (LM).
- ✓ Right Most Maximum (RM).

(f) Post-processing

Output scaling is also relevant. In case the output is defined on a standard universe this must be scaled to engineering unit, for instance, volts, meters, or tons per hour.

An example is the scaling from the standard universe (-1, 1) to the physical units (-5, 5) volts.

2.5.3 Application of FLC

The description of the technological process is available only in word form, not in analytical form.

- It is not possible to identify the parameters of the process with precision.
- The description of the process is too complex and it is more reasonable to express its description in plain language words.
- The controlled technological process has a "fuzzy" character, i.e. its behavior is not fully unequivocal under precisely defined conditions [16].

2.5.4 Advantages of using fuzzy technique

- Simplicity of control and smooth operation.
- High degree of tolerance.
- Low cost.
- Reduce the effect of non-linearity.
- Inherent approximation capability.
- Possibility to design without knowing the exact mathematical model of the process.

2.6 Neuro Fuzzy System

In spite of the advantages in fuzzy control, the main limitations are the lack of a systematic design methodology and the difficulty in predicting stability and robustness of the controlled system. A trial-and-error iterative approach is taken for the controller design due to which we get sluggish response. The neuro-fuzzy learning incorporates the architecture of neural network based fuzzy inference system. A given training data set is partitioned into a set of clusters based on subtractive clustering method. This is fast and robust method to generate the suitable initial membership functions and rule base. A fuzzy if-then rule is then extracted from each cluster to form a fuzzy rule base from which a fuzzy neural network is designed. Then a hybrid learning algorithm is used to refine the parameters of fuzzy rule base.

2.6.3 Adaptive neuro fuzzy inference system

Adaptive Neuro-Fuzzy Inference System (ANFIS) is one of the most successful schemes which combine of the fuzzy logic and neural network systems into a single technique. (Jang, 1993). An ANFIS works by applying neural learning rules to identify and tune the parameters and structure of a Fuzzy Inference System (FIS). There are several features of the ANFIS which enable it to achieve great success in a wide range of scientific applications. The attractive features of an ANFIS include: easy to implement, fast and accurate learning, strong generalization abilities, excellent explanation facilities through fuzzy rules, and easy to incorporate both linguistic and numeric knowledge for problem solving (Jang & Sun, 1995; Jang et al., 1997). According to the neuro-fuzzy approach, a neural network is proposed to implement the fuzzy system, so that structure and parameter identification of the fuzzy rule base are accomplished by defining, adapting and optimizing the topology and the parameters of the corresponding neuro-fuzzy network, based only on the available data.

By using the neuro-fuzzy scheme, the fuzzy inference system can be tuned with a neural network algorithm based on some collection of input-output data, which then allow the fuzzy system to learn. Fundamentally, it takes a FIS and tunes it with a backpropagation algorithm based on some collection of input-output data. This allows FIS to learn. The network structure facilitates the computation of the descendent gradient vector for parameters in a FIS. Once the gradient vector is obtained, we can apply a number of optimization routines to reduce an error measure (usually defined by the sum of the squared difference between actual and desired outputs) [17].

2.6.4 Neuro-fuzzy controller

The adaptive neuro-fuzzy inference system uses a feedforward network to search for fuzzy decision rules that perform well in a given task. Using a given input/output data set ANFIS creates a fuzzy inference system for which membership function parameters are adjusted using a combination of a back propagation and least square method [18].

The proposed scheme utilizes sugeno-type FIS controller, with the parameters inside the FIS decided by the neural-network back propagation method. The ANFIS is designed by taking speed error (e) and change in speed error (ce) as the inputs. The output stabilizing signals is computed using the fuzzy membership functions depending on these variables. ANFIS-editor is used for realizing the system and implementation. In a conventional fuzzy approach the membership functions and the consequent models are fixed by the model designer according to a prior knowledge. If this set is not available but a set of input-output data is observed from the process, the components of a fuzzy system (membership and consequent models) can be represented in a parametric form and the parameters are tuned by neural networks. In that case the fuzzy systems turn into neuro-fuzzy system. A fuzzy system can explain the knowledge it encodes but can’t learn or adapt its knowledge from training examples, while a neural network can learn from training examples but cannot explain what it has learned. Fuzzy systems and neural networks have complementary strengths and weaknesses. As a result, many researchers are trying to integrate these two schemes to generate hybrid models that can take advantage of strong points of both.

CHAPTER THREE

CONTROL SYSTEM DESIGN OF DC MOTOR

CHAPTER THREE

CONTROL SYSTEM DESIGN OF DC MOTOR

3.1 Introduction

In spite of the development of power electronics resources, the direct current machine became more and more useful. Nowadays their uses isn't limited in the car applications (electrics vehicle), in applications of weak power using battery system (motor of toy) or for the electric traction in the multi-machine systems too. The speed of DC motor can be adjusted to a great extent as to provide controllability easy and high performance. The controllers of the speed that are conceived for goal to control the speed of DC motor to execute one variety of tasks, is of several conventional and numeric controller. The main idea of this thesis is to develop ANFIS controller, FLC and conventional PID controller shown in Figure 3.1 by using MATLAB/SIMULINK. The target of the study is control the speed of the DC motor. The results for all simulations are elaborate and discuss in chapter four in result section.

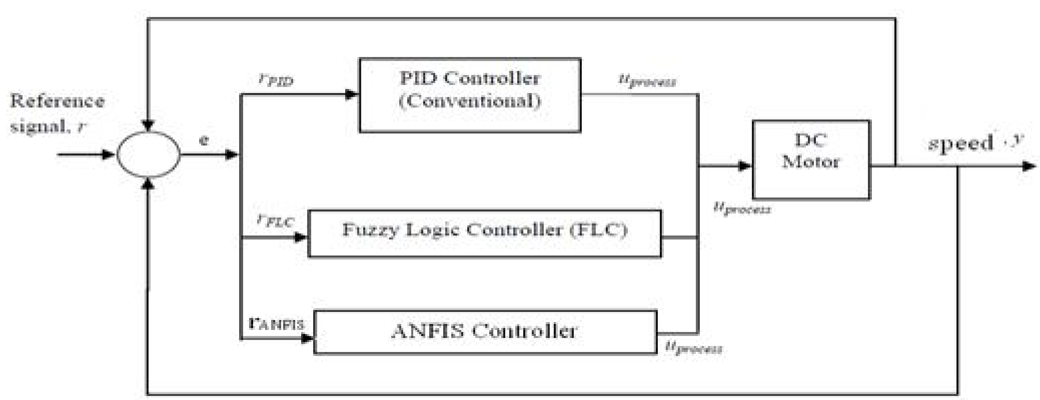

Figure 3.1 Speed control of DC motor

Main controller of this project is to perform neuro fuzzy controller while fuzzy controller and PID controller is use to compare the performance of the controller.

3.2 Mathematical Model of PMDC Motor

The goal of this section is to develop the mathematical model in sense that is related to the applied voltage to the armature to the velocity of motor. By considering electrical a mechanical characteristics of the system so as to balance equation developed. Modeling can be defined as simplified representation of a system, for example using mathematical equation. But a variety of mathematical models for direct drive cannot give the exact and definite description of drive behavior. The approach to mathematical model constructing and its verification is described. The model which with sufficient accuracy describes direct drive is based on model building of physical processes which take place in electrical drive. Mathematical model presented is used to compute simulation of control system with different regulators to improve motor's dynamics, and, as the result, hardware performance and precision. In this chapter DC motor speed control system is described. The DC motor used is of permanent magnet type. The model of the permanent magnet DC motor is first developed, followed by the discussion on the mechanism that will be used to drive the motor.

A permanent magnet DC motor consists of permanent magnet stator and armature windings in the rotor. The armature winding is supplied with a DC voltage that causes a DC current to flow in the windings. Interaction between the magnetic field produced by the armature current and that of the permanent magnet stator causes the rotor to rotate. The equivalent circuit of a PMDC motor is shown in Figure 3.2. The controller is selected so that the error between the system output and reference signal eventually tends to its minimum value, ideally zero. There are various DC motor types. Depending on type, a DC motor may be controlled by varying the input voltage whilst another motor only by changing the current input.

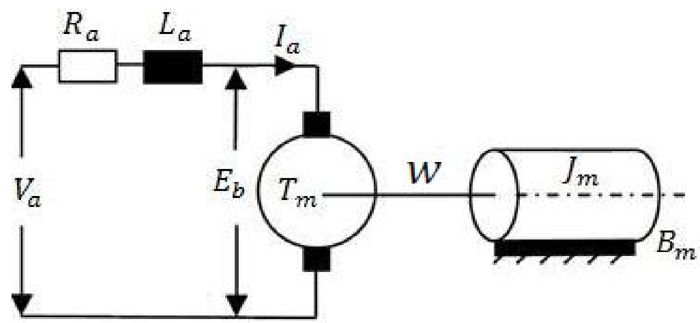

Figure 3.2: The equivalent circuit of a PMDC motor

The equations describing the characteristic of a PMDC motor can be determined from Figure 3.2. The basic structure of a DC motor can be divided into two parts; the voltage controlled circuitry and mechanical rotor. The motor torque provides the movement over inertia for the rotor system. If the motor initially is in a static condition, it requires a larger torque value to start the motor. If the motor initially is in a moving condition, the inertia of the motor will sum up with the system torque to give a grater rotation, which will further increase the moment speed of this system.

The characteristic equations of the PMDC motor are represented as:

(a) Electrical equations

The electrical circuit of motor is shown in Figure 3.2 represent a series wound DC motor. The selection a series wound DC motor is better because this type of motor provide better performance on heavy torque loads. Figure 3.2 shows the voltage source V_a across the coil of armature. The electrical circuit equivalent of the armature and inductance L_a and resistance R_a are in series with induced voltage $E_g$which opposes the source voltage. The induced voltage is generated due to electrical coil rotating in fixed flux lines of the permanent magnet .This voltage is often called back-emf voltage. Using the Kirchoff's law, the circuit equivalent equation can be developed as shown in equation (3.1) that needs to be taken into consideration is the DC motor equivalent circuit characteristic equation:

$$v_a(t) = R_a.i_a(t) + L_a.\frac{di_a(t)}{dt} + e_b(t) \tag{3.1}$$

(b) Mechanical equation

The turning effect of force called torque or a force are tries to rotate about its own axis is called torque. Here different torque is effecting the motors and its order to get mechanical equation take all the torque equal to zero. The current which passes through the armature winding is proportional to the electromagnetic torque which can be expressed as:

$$T_m(t) = K_T i_a(t) \tag{3.2}$$

The developed torque must be equal to the load torque:

$$T_m(t) = J_m.\frac{d\omega(t)}{dt} + B_m.\omega(t) + T_L(t) \tag{3.3}$$

The back-emf is related to the rotational velocity by the constant factor expressed as:

$$e_b(t) = K_b.\omega(t) \tag{3.4}$$

$$v_a(s) = R_a.i_a(t) + L_a.i_a(t) + K_b.\omega(t) \tag{3.5}$$

$$k_T.i_a(t) = J_m.\frac{d\omega(t)}{dt} + B_m.\omega(t) + T_L(t) \tag{3.6}$$

Laplace transform of (3.5) and (3.6) are:

$$V_a(s) = R_a.I_a(s) + s.L_a.I_a(s) + K_b.\omega(s) \tag{3.7}$$

$$k_T.I_a(s) = s.J_m.w(s) + B_m.\omega(s) + T_L(s) \tag{3.8}$$

If current is obtained from (3.8) and substitute in (3.7) we have:

$$V_a(s) = \omega(s).\frac{1}{K_T}[R_a.I_a(s) + s.L_a.I_a(s) + K_b.\omega(s)] \tag{3.9}$$

The relation between rotor shaft speed and applied armature voltage is represented by transfer function

$$\frac{\omega(s)}{V(s)} = \frac{K_T}{(L_a s+R_a)(J_m s+B_m)+K_T K_b} \tag{3.10}$$

Where:

I_a= armature current (A)

V_a= armature voltage (V)

E_b= back-emf voltage (V)

K_b= back-emf constant (V/rad/s)

L_a= armature circuit inductance (H)

R_a= armature circuit resistance (Ω)

ω = motor speed (rad/s)

J_m= moment of inertia of (load and rotor) (N.m^2)

B_m = viscous friction constant (N.m/rad/s)

K_T= torque constant (N.m/A)

T_L= load torque (N-m)

T_m = motor developed torque (N-m)

K_b=back emf constant

It can be noticed from equation (3.9) that the motor speed can be varied by controlling the armature voltage V_a or the armature current I_a. The direction of rotation of the motor can be reversed by reversing the polarity of the voltage applied to the terminal**.** If the voltage applied to the motor terminal is reversed, the direction of current flowing in the armature winding is also reversed. This will cause the motor to produce torque in the reverse direction. Note here that the polarity of the back-emf voltage is also reversed. The equation (3.10) can be used in order to obtain the SIMULINK model

(c) Parameters of PMDC motor

The DC motor parameters considered for this work is that used in an undergraduate experiment carried out in Carnegie Mellon Control Laboratory of University of Michigan, United States [6].

R_a = 1Ω

L_a= 0.5H

K_b= 0.01 V/rad/s

J_m= 0.01 kg.m2

B_m = 0.001 N.m/rad/s

K_T = 0.01Nm/Amp

3.3 SIMULINK Model of PMDC Motor

SIMULINK is a software package for modelling, simulating, and analyzing dynamic systems. It supports linear and nonlinear systems, modelled in continuous time, sampled time, or a hybrid of the two. Systems can also be multi rate, i.e., have different parts that are sampled or updated at different rates. SIMULINK encourages user to try things out. The user can easily build models from scratch, or take an existing model and add to it. Simulations are interactive, so the user can change parameters on the fly and immediately see what happens. The users have instant access to all the analysis tools in MATLAB, so they can take the results and analyze and visualize them. A goal of SIMULINK is to give user a sense of the fun of modelling and simulation, through an environment that encourages user to pose a question, model it, and see what happens. With SIMULINK, user can move beyond idealized linear models to explore more realistic nonlinear models, factoring in friction, air resistance, gear slippage, hard stops, and the other things that describe real-world phenomena.The MATLAB/SIMULINK model of the system under study is shown in Figure 3.3.

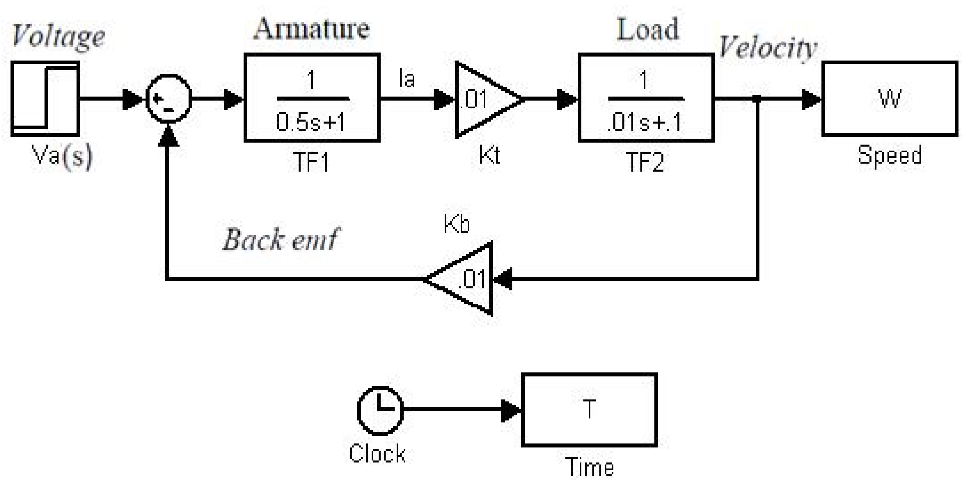

Figure 3.3: SIMULINK model for PMDC motor

3.4 Design of PID Controller

The combination of Proportional, Integral and Derivative control action is called PID controller action. The PID controller is one of the conventional controllers and it has been widely used for the speed control of DC motor drive. The parameters values of PID can be interpreted in terms of time, P depends on the present error, I on the accumulation of past errors, and D is a prediction of future errors based on current rate of change. The weighted sum of these three actions is used to adjust the process via a control element such as the speed. The conventional method has the following difficulties:

i. It depends on the accuracy of the mathematical model of the systems.
ii. The expected performance is not met due to the load disturbance, motor saturation and thermal variations.
iii. Classical linear control shows good performance only at one operating speed.
iv. The coefficients must be chosen properly for acceptable results, whereas choosing the proper coefficient with varying parameters like set point is very difficult.

To implement conventional control, the model of the controlled system must be known. The usual method of computation of mathematical model of a system is difficult. When there are system parameter variations or environmental disturbance, the behavior of the system is not satisfactory. Usually classical control is used in electrical motor drives. The classical controller designed for high performance increases the complexity of the design and hence the cost.

The PID controller is placed in the forward path, so that its output becomes the voltage applied to the motor armature the feedback signal is a velocity. The output velocity signal w(t) is summed with a reference or command signal r(t) to form the error signal e(t). Finally, the error signal is the input to the PID controller.

3.4.3 PID parameters

PID controller can be investigated under three main categories. Each controller has different properties in terms of controlling the whole system.

- In proportional control, adjustments are based on the current difference between the actual and desired speed.
- In integral control, adjustments are based on recent errors.
- In derivative control, adjustments are based on the rate of change of errors.

3.4.4 Tuning PID parameters

There are many tuning methods, but most common methods are as follows:

✓ Manual tuning method.

✓ Ziegler-Nichols tuning method.

✓ Cohen-Coon tuning method.

✓ PID tuning software methods (ex. MATLAB).

PID controllers are usually tuned using hand tuning or Ziegler –Nicholas methods and soft tuning. Hand tuning or Ziegler Nichols is generally used by experienced control engineers based on the rules shown in Table 2.1. But these rules are not always valid. For example if an integrator exists in the plant, then increasing K_P results in a more stable control. Figure 3.11 shows the SIMULINK model for speed control of PMDC motor using PID controller.

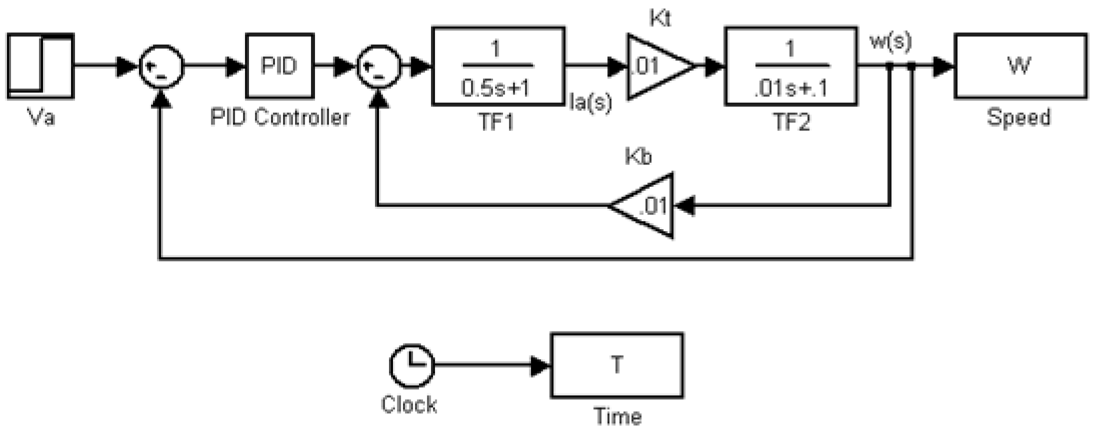

Figure 3.4:SIMULINK model for speed control of PMDC motor using PID

3.5 Fuzzy Controller Design

Fuzzy control provides a formal methodology for representing, manipulating and implementing a human's heuristic knowledge of how to control a system. Fuzzy control system design essentially amounts to:

1. Choosing the fuzzy controller inputs and outputs.
2. Choosing the preprocessing that is needed for the controller inputs and possibly post processing that is needed for the outputs.
3. Designing each of the four components of the fuzzy controller as shown in

Moreover, most often the designer settles on an inference mechanism and may use this for many different processes. Hence, the main part of the fuzzy controller that we focus on for design is the rule-base. The rule-base is constructed so that it represents a human expert "in-the-loop". Hence, the information that we load into the rules in the rule-base may come from an actual human expert who has spent long time learning how best to control the process. In other situations there is no such human expert, and the control engineer will simply study the plant dynamics (perhaps using modeling and simulation) and write down a set of control rules that makes sense. The fuzzy controller has four main components:

i. The "rule-base" holds the knowledge, in the form of a set of rules, of how best to control the system.

ii. The inference mechanism evaluates which control rules are relevant at the current time and then decides what the input to the plant should be.

iii. The fuzzification interface simply modifies the inputs so that they can be interpreted and compared to the rules in the rule-base.

iv. Defuzzification converts the conclusions reached by the inference mechanism into plant inputs.

3.5.3 Developing fuzzy expert system

Define linguistic variables and also specify the problem is the first step to make the mamdani controller.

(a) Linguistic variables and their representations

Linguistic variable (linguistic term) defined by Zadeh, a linguistic variable are mean a variable whose values are words or sentences in a natural or artificial language. They are three main linguistic variables:

i. error (e)

ii. change in error (ce)

iii. output (u)

In fuzzy logic, some of the typical linguistic terms used are shown in Table 3.1

Table 3.1 Meaning of typical linguistic term in fuzzy logic

Linguistic term	Meaning
PB	Positive Big
PS	Positive Small
ZR	Zero

NS	Negative Small
NB	Negative Big

(b) Fuzzification

The fuzzy set of the error ‘e’ input which contains five triangular memberships, the fuzzy set of the change error ’ce’ input which contains five triangular memberships and the fuzzy set of the output ‘u’ which contains five triangular memberships are shown in Figure 3.5.

(c) Defuzzification

Bisector of Area (BOA) method was used in this study.

(d) Control base rules

Table 3.2 presents the knowledge base defining the rules for the desired relationship between the input and output.

Table 3.2: Rule base for five membership functions

e / ce	**NB**	**NS**	**ZR**	**PS**	**PL**
NB	NB	NB	NB	NS	ZR
NS	NB	NS	NS	ZR	PS
ZR	NB	NS	ZR	PS	PB
PS	NS	ZR	PS	PS	PB
PB	ZR	PS	PB	PB	PB

3.5.4 Fuzzy logic toolbox

There are five primary GUIs tools for building, editing and observing fuzzy inference systems in the toolbox:

i. Fuzzy inference system editor.

ii. Membership function editor.

iii. Rule editor.

iv. Rule viewer.

v. Surface viewer.

These GUIs are dynamically linked and if the changes make to the FIS to one of the toolbox, the effect can be seen in other GUIs. In addition to these five primary GUIs, the toolbox includes the graphical ANFIS editor GUI, which is used for building and analyzing Sugeno-types adaptive neural fuzzy inference systems.

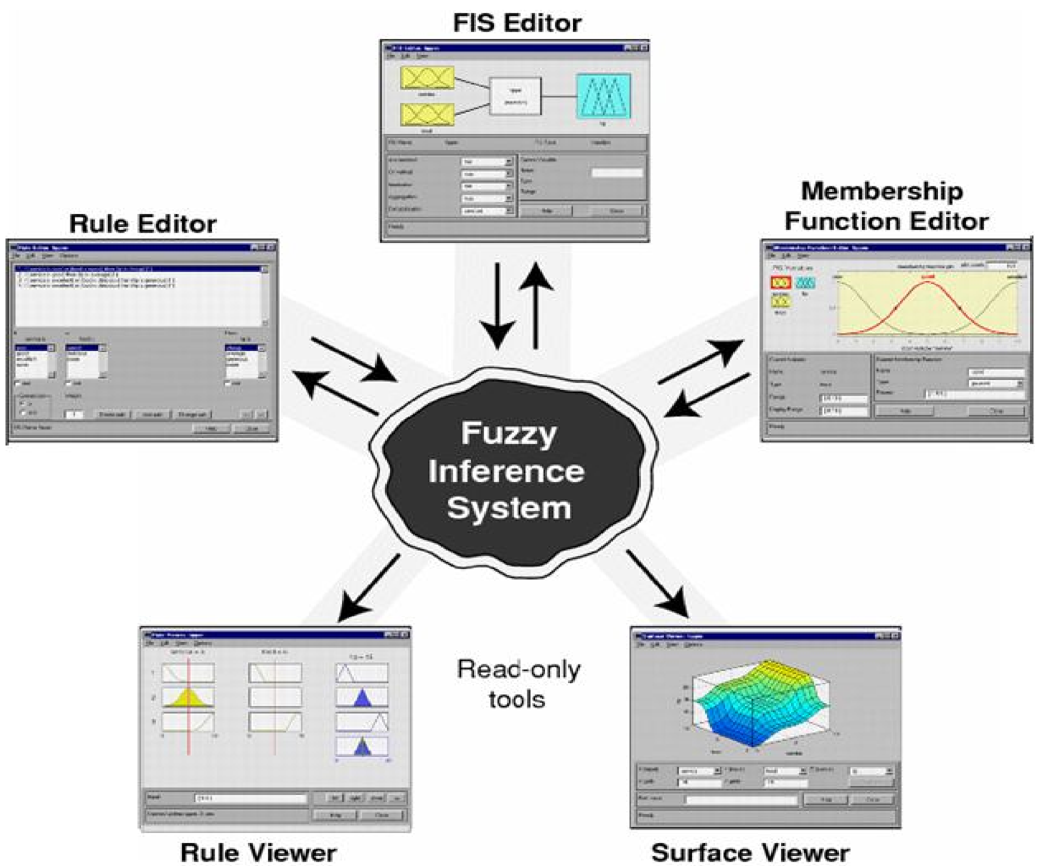

Figure 3.5: Fuzzy inference system

• **Fuzzy basic FIS editor**

The FIS editor displays high-level information about a fuzzy inference system shown in Figure 3.6. Fuzzy logic toolbox does not limit at the top is a diagram of the system with each input variable on the left and the output on the right clearly labeled. The following step shown how to open the FIS editor:

i. To start the system from scratch, type fuzzy at the MATLAB prompt.

ii. Select edit > add variable > input (to add second input).

iii. Click the yellow box input1. This box is highlighted with a red outline.

iv. edit the name field from input1 to 'e', and press enter.

v. Click the yellow box input2. This box is highlighted with a red outline.

vi. Edit the name field from input2 to' ce', and press enter.

vii. Click the blue box output1.

viii. Edit the name field from output1 to 'u', and press enter.

ix. Select file > export > to workspace.

x. Enter the workspace variable name "SIDDIG', and click ok.

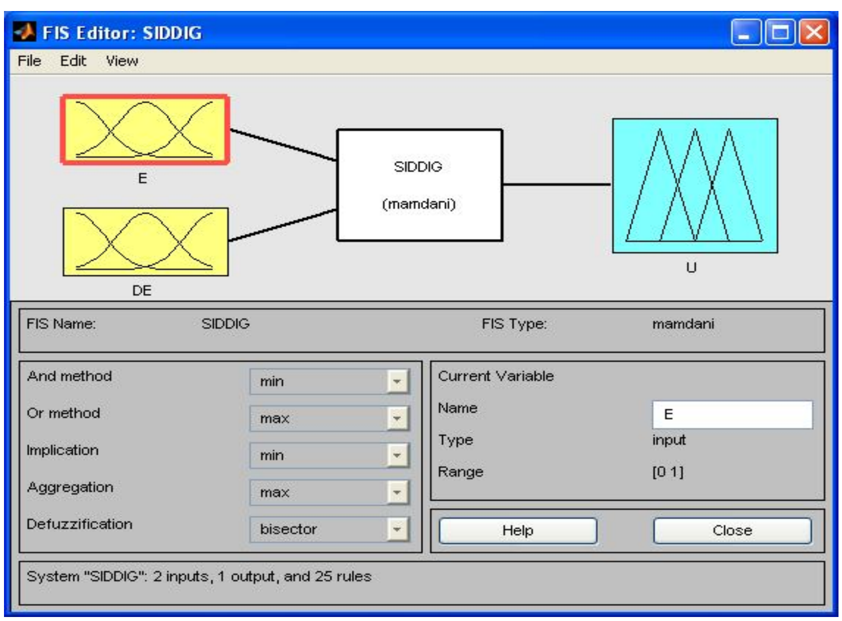

Figure 3.6: FIS editor

By adding input variable from edit menu then FIS give two inputs variable which one is error (e) and another is change of error (ce) eatch one consist of five membership functions to give 25 rules.

- **Membership function editor**

The membership function editor shares some features with the FIS editor. The membership function editor is the tool that lets the programmer displays and edits all of the membership functions associated with all inputs and output variables for entire fuzzy inference system. The step below shows how to open the membership function editor:

✓ Within the FIS editor windows, select edit > membership functions.

✓ Within the FIS editor, double click the blue icon called 'SIDDIG'.

✓ At the command line, type mfedit ('SIDDIG').

The mfedit ('SIDDIG') generates a membership function editor shown in Figure 3.7 that allows to modify all the membership functions for the FIS stored in the file. The Membership Functions (MFs) editor is used to create, remove, and modify the MFs for a given fuzzy system. On the left side of the diagram is a "FIS variable" region that used to select the current variable by clicking once on one of the displayed boxes. Information about the current variable is displayed in the text region below the palette area. To the right is a plot of all the MFs for the current variable. It could select any of these by clicking once on the line or name of the MF. Once selected, It could modify the properties of the MF using the controls in the lower right. MFs are added and removed using the edit menu.

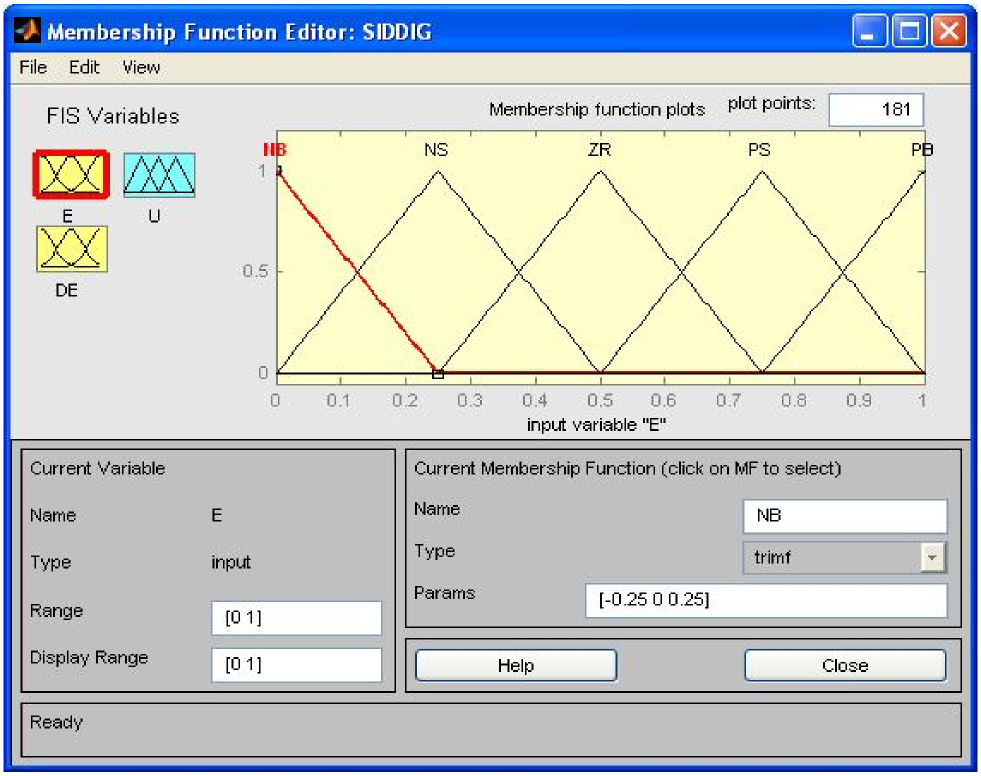

Figure 3.7: Membership function

• **Rule editor**

The rule editor shown in Figure 3.8, when invoked using ruleedit ('SIDDIG'), is used to modify the rules of a FIS structure stored in a file, SIDDIG.fis. It can also be used to inspect the rules being used by 'SIDDIG' fuzzy inference system. Based on the description of the input and output variable defined with the FIS editor, the rule editor allows constructing the rule statements automatically. From GUI:

✓ Create rules by selecting an item in each input and output variable box and one connection item and clicking add rule. You can choose none as one of the variable qualities to exclude that variable from a given rule and choose not under any variable name to negate the associated quality.

✓ Delete a rule by selecting the rule and clicking delete rule.

✓ Edit a rule by changing the selection in the variable box and clicking change rule.

✓ Specify weight to a rule by typing in a desired number between 0 and 1 in Weight. If you do not specify the weight, it is assumed to be unity (1).

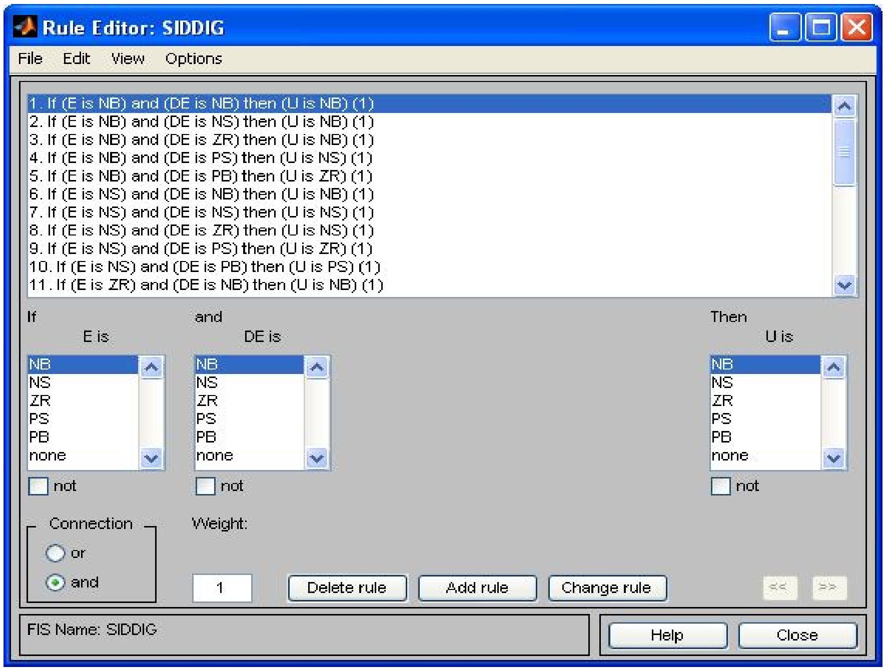

Figure 3.8: Rule editor

- **Rule viewer**

The rule viewer displays shown in Figure 3.9 in one screen, all parts of the fuzzy inference process from inputs to outputs.

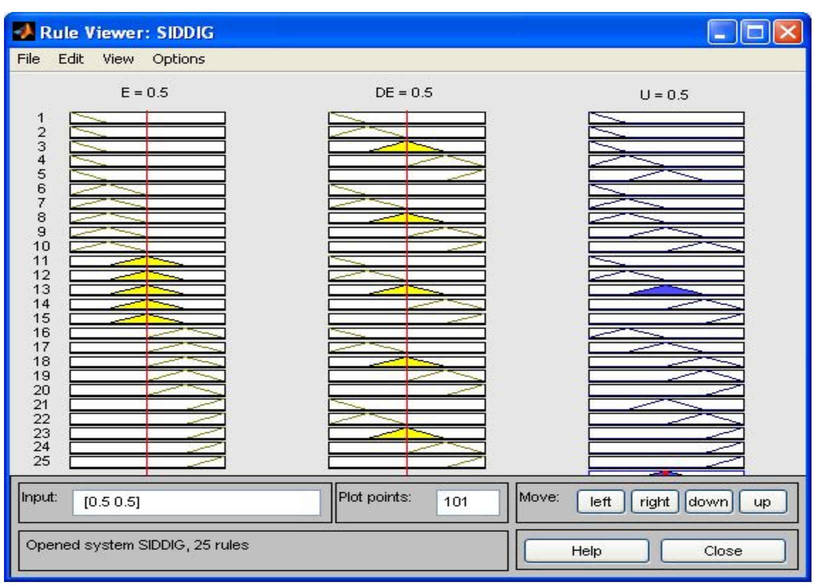

Figure 3.9: Rule viewer

Each row of plots corresponds to one rule, and each column of plots corresponds to either an input variable (yellow, on the left) or an output variable (blue, on the right). It could change the system input either by typing a specific value into the Input window or by moving the long yellow index lines that go down each input variable's column of plots.

- **Output surface viewer**

The input-output mapping can be observed by viewing surface. Choose view menu and under it view surface. The surface viewer shown in Figure (3.10) invoked using surfview ('SIDDIG') is a GUI tool that lets to examine the output surface of a FIS, SIDDIG.fis, for any one or two inputs. Since it does not alter the fuzzy system or its associated FIS matrix in any way, it is a read-only editor. It is clear that our map is nonlinear. This is where the power of fuzzy systems is strong.

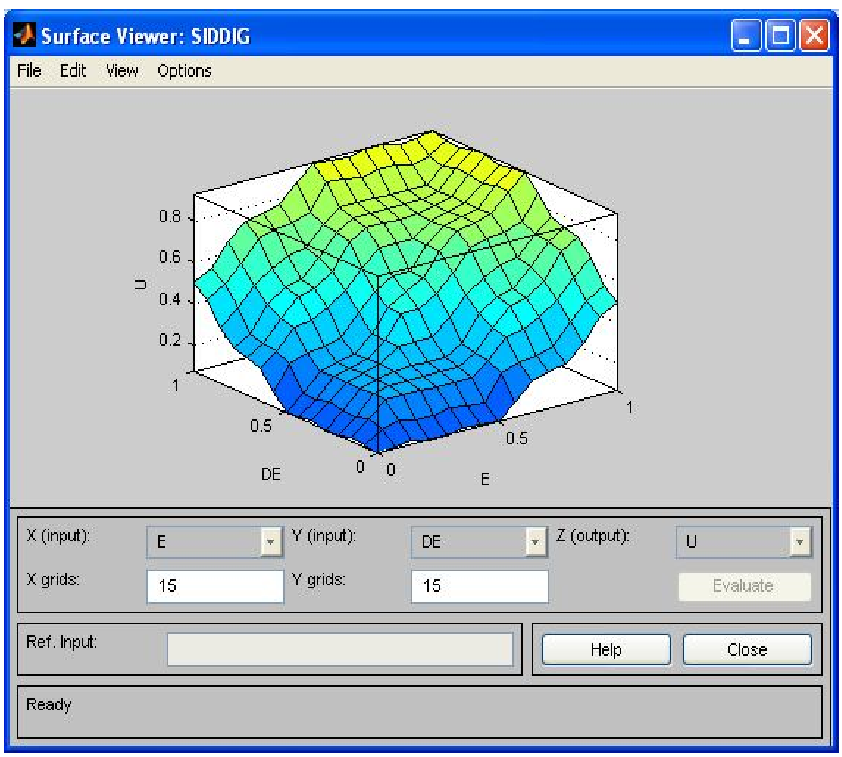

Figure 3.10: Surface viewer

Figure 3.11 shows the SIMULINK model for speed control of PMDC motor using FLC.

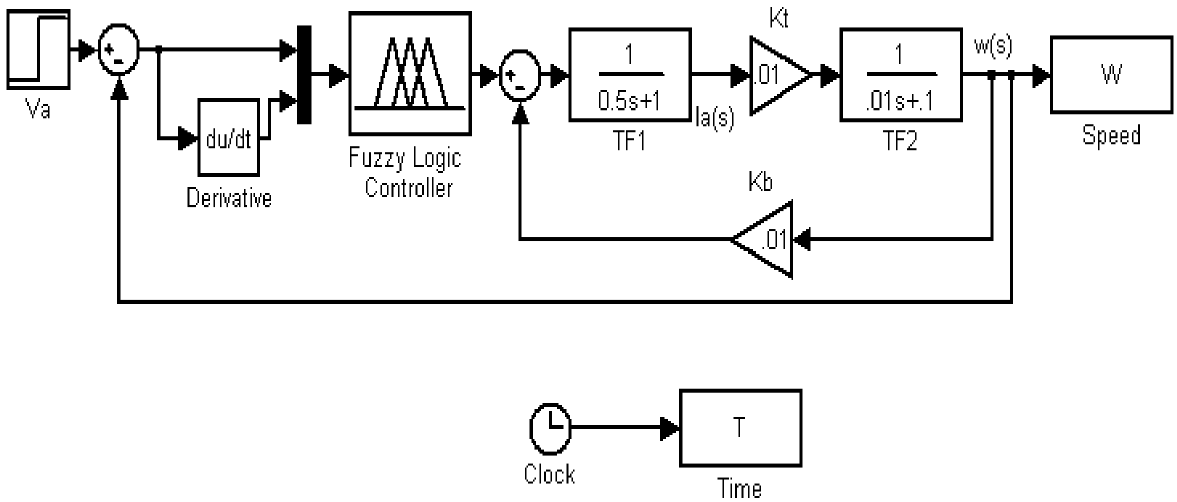

Figure 3.11 SIMULINK model for speed control of PMDC motor using FLC

3.6 Adaptive Neuro Fuzzy Controller Design

ANFIS uses a hybrid learning algorithm to identify the membership function parameters of single-output, Sugeno type fuzzy inference systems. A combination of least-squares and backpropagation gradient descent methods are used for training FIS membership function parameters to model a given set of input/output data. The ANFIS uses a feedforward network to search for fuzzy decision rules that perform well in a given task. Using a given input/output data set ANFIS creates a fuzzy inference system for which membership function parameters are adjusted using a combination of a back propagation and least square method.

ANFIS estimator design consists of two parts: constructing and training. In the constructing part, structure parameters are determined. These are type and number of input MFs, and type of output MFs. Effective partition of the input space is important and it can decrease the rule number and thus increase the speed in both learning and application phase. Output MFs can be either a constant or in linear form. Both of the two forms are used for the output MF in this study. Having described the number and type of input MFs, the estimator rule base is constituted. Since, there is no standard method to utilize the expert knowledge; automatic rule generation method is usually preferred. According to this method, for instance, an ANFIS model with two inputs and five MFs on each input would result in 52 = 25 Takagi-Sugeno fuzzy if-then rules automatically. Although this method may requires much computational knowledge, especially in systems that have to be defined with many inputs. It is used in this study due to the advantage of MATLAB software. Therefore, rule bases of estimators are formed automatically with the number of inputs and number of MFs. After the ANFIS structure is constructed, learning algorithm and training parameters are chosed. As mentioned earlier in this paper, back propagation or hybrid learning can be used as a learning algorithm. Therefore, hybrid learning algorithm is adopted in this study. Parameters in the algorithm are epoch size (presentation of the entire data set), error tolerance, initial step size, step size decrease rate, and step size increase rate. Since there is no exact method in the literature to find the optimum of these parameters a trial and error procedure is used. MATLAB fuzzy logic toolbox is used to design ANFIS estimators' structures. Using the given training data set, the toolbox constructs an ANFIS structure using either a back propagation algorithm alone, or in combination with least squares type of method (hybrid algorithm). ANFIS model can be generated either from the command line, or through the ANFIS editor GUI. In this study, ANFIS editor GUIs is used to generate ANFIS models with the chosen design parameters in the construction phase.

As it was mentioned, for good performance of an ANFIS controller, it should be fed by optimized inputs and outputs. In order to achieve this goal, many methods have been invented. This research tries to create a method of feeding the inputs and output of an ANFIS which is here called PID-based (composed method) ANFIS method in five steps:

(1) In the first step of this method we assume a special transfer function and control it with a PID controller with the best P, I and D parameters. This controller should control that process well with best P I and D parameters.
(2) At this stage of this method, the output of the PID controller should be sent to the workspace in MATLAB software in some points. It's better to have some more points to have better accuracy.
(3) At this step, these points should be modified and optimized the points by an expert operator, on the points that the PID controller does not act well.
(4) During this step of this method after modification the points, we train the ANFIS inputs and output with these modified points. The inputs are the error and the deviation of the error, and the output is the output of third step.
(5) In this step use the new ANFIS controller to control the mentioned transfer function.

After above steps the ANFIS controller has some fuzzy rules and memberships with special inputs and output. This may used also for unknown non-linear process plants. In the first step ,In order to achieve this goal to control the process with a PID controller. After that with the composed method we can have the ANFIS controller to control that unknown process much better. The four steps of ANFIS estimator design are as follows:

1. Generated training data is loaded to the Editor GUI.

2. Design parameters, number and type of input and output MFs, are chosen. Thus, initial ANFIS structure is formed.

3. The code for training is run with an initial structure.

4. ANFIS structure constituted after training is saved.

3.6.1 ANFIS editor

Using anfisedit to bring up the ANFIS editor shown in Figure 3.12 GUIs from which the data set and train anfis were loaded. The ANFIS editor GUI invoked using anfisedit('a'), opens the ANFIS editor GUI from implement ANFIS using a FIS structure stored as a file a.FIS anfisedit(a) operates the same way for a FIS structure a, stored as a variable in the MATLAB workspace, Figure 3.13 shows ANFIS model structure . On the ANFIS editor GUI, there is a menu bar that allows you to open related GUI tools, open and save systems, and so on. The file menu is the same as the one found on the FIS editor. By using the following edit menu item:

✓ Undo to undo the most recent change.

✓ FIS properties to invoke the FIS editor.

✓ Membership functions to invoke the membership function editor.

✓ Rules to invoke the rule editor.

✓ By using the following view menu items:

✓ Rules to invoke the rule viewer.

✓ Surface to invoke the surface viewer.

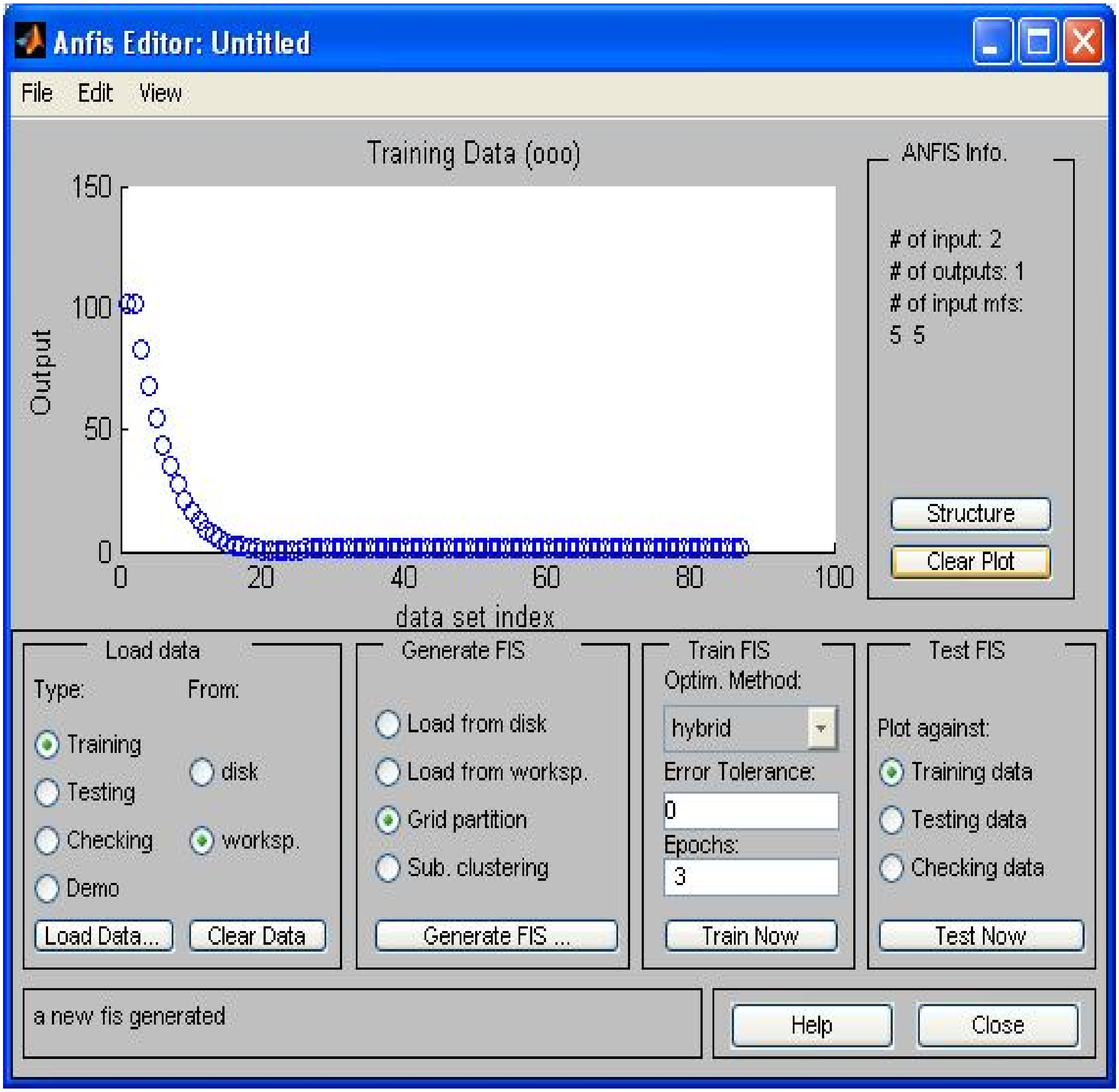

Figure 3.12: ANFIS editor

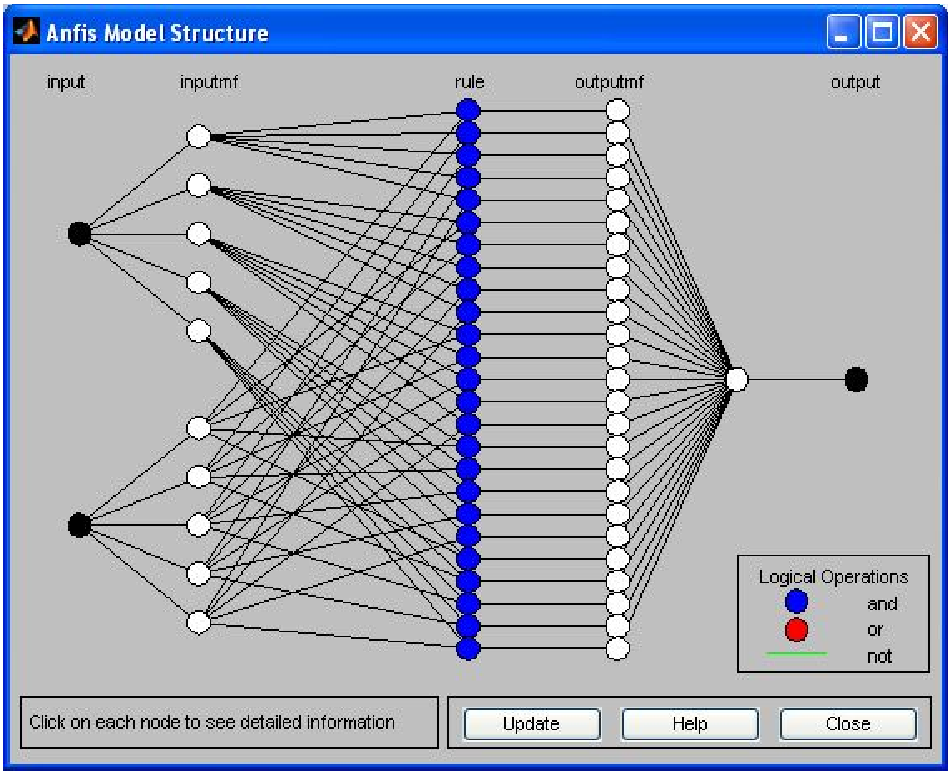

Figure 3.13: ANFIS model structure

Figure 3.14 shows the SIMULINK model for speed control of PMDC motor using ANFIS controller.

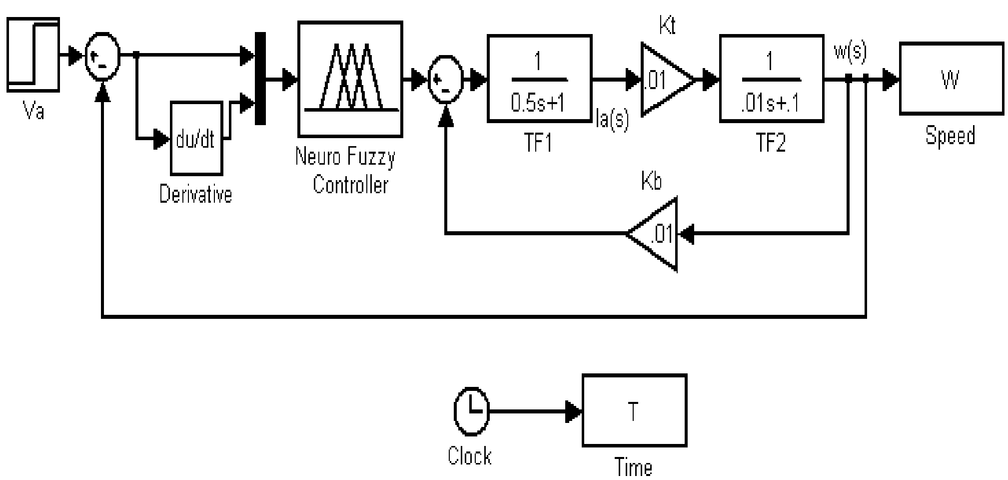

Figure 3.14: SIMULINK model for speed control of DC motor using ANFIS

3.6.2 Flow chart for ANFIS training process

The ANFIS training off-line methodology using ANFIS GUI in MATLAB fuzzy logic Toolbox is summarized in Figure 3.15. The process begins by obtaining a training data set (input/output data pairs). The training data set is used to find the premise parameters for the membership functions. A threshold value for the error between the actual and desired output is determined. The consequent parameters are found using the least-squares method. Then an error for each data pair is found. If this error is larger than the threshold value, then the premise parameters are updated using the gradient decent method (backpropagation).The process is terminated when the error becomes less than the threshold value. Then the checking data set is used to compare the model with actual system.

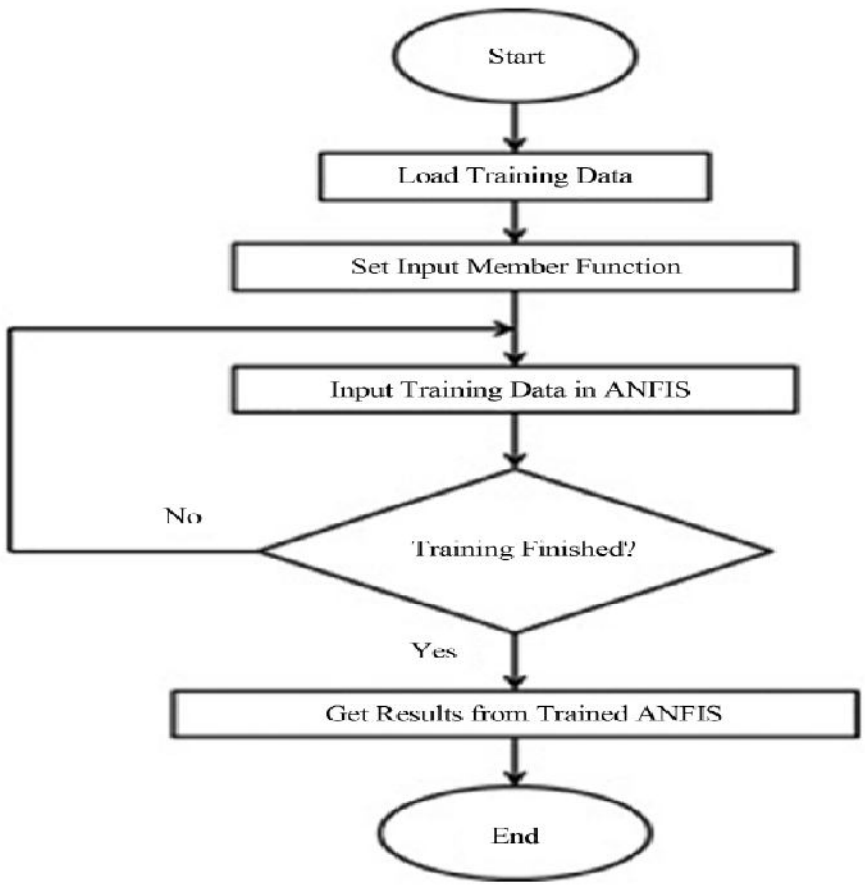

Figure 3.15: ANFIS training proces

CHAPTER FOUR

SIMULATION RESULTS AND DISCUSSION

CHAPTER FOUR

SIMULATION RESULTS AND DISCUSSION

4.1 Simulation Results

In this thesis studied about different method for speed control of PMDC motor. In order to validate the control strategies as described above, digital simulation were carried out on a PMDC motor drive system whose parameters are given in chapter three. We applied step input to three types of control systems (Fuzzy, PID and ANFIS).

The open loop transfer function behavior of the PMDC motor to a unit step response is shown in Figure 4.1. It could be observed that the motor response to a unit step input signal, that is, an equivalent of 1V supply voltage is 0.1 rad/sec. This is one-tenth of the desired response. Also, the settling time is 3s of which a reduction is sought. Steady state error could also be improved. The response of motor speed with PID controller is shown in Figure 4.2. Figure 4.3 shown response of motor speed to a unit step input with a fuzzy controller. Figure 4.4 shows the response of the motor speed to a unit step input with a neuro-fuzzy controller. Figure 4.5 shows the response of motor speed to a unit step input with three different controllers from which it is clear that ANFIS controller performs slightly better than the other two controllers.

4.2 Discussion

Comparing the fuzzy and neuro-fuzzy controllers, the results show a slight change as shown in Figure. 4.3 and Figure. 4.4. In spite of the advantages in fuzzy control, the main limitations are the lack of a systematic design methodology and the difficulty in predicting stability and robustness of the controlled system. A trial-and-error iterative approach is taken for the controller design due to which we get sluggish response. The neuro-fuzzy learning incorporates the architecture of neural network based fuzzy inference system. A given training data set is partitioned into a set of clusters based on subtractive clustering method. This is fast and robust method to generate the suitable initial membership functions and rule base. A fuzzy if-then rule is then extracted from each cluster to form a fuzzy rule base from which a fuzzy neural network is designed. Then a hybrid learning algorithm is used to refine the parameters of fuzzy rule base.

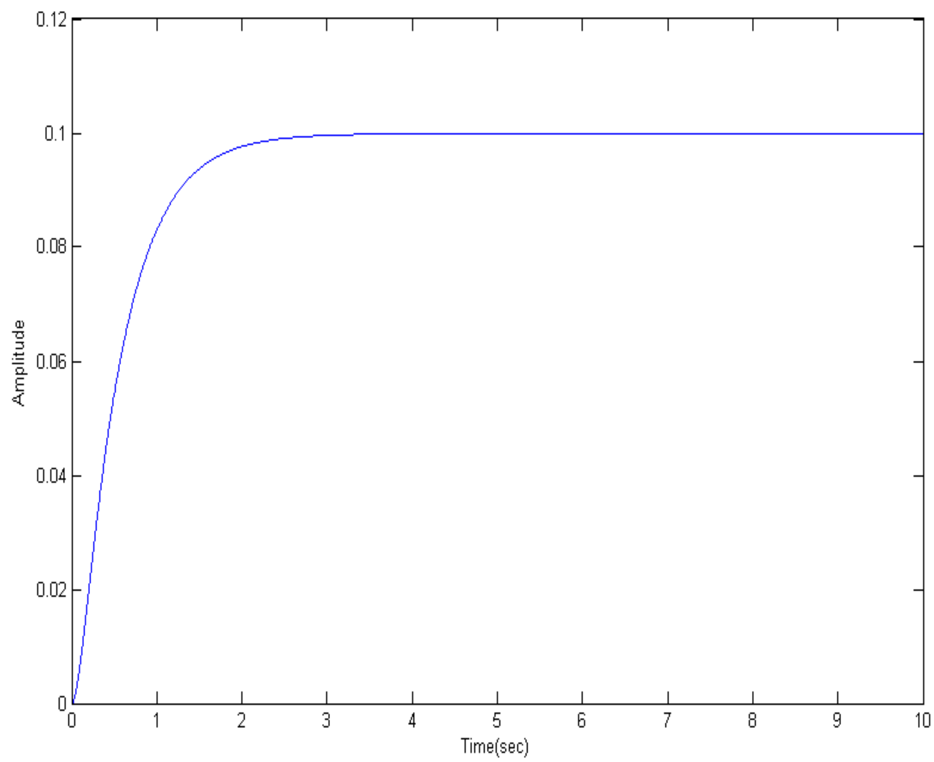

Figure 4.1: Motor speed response without controller

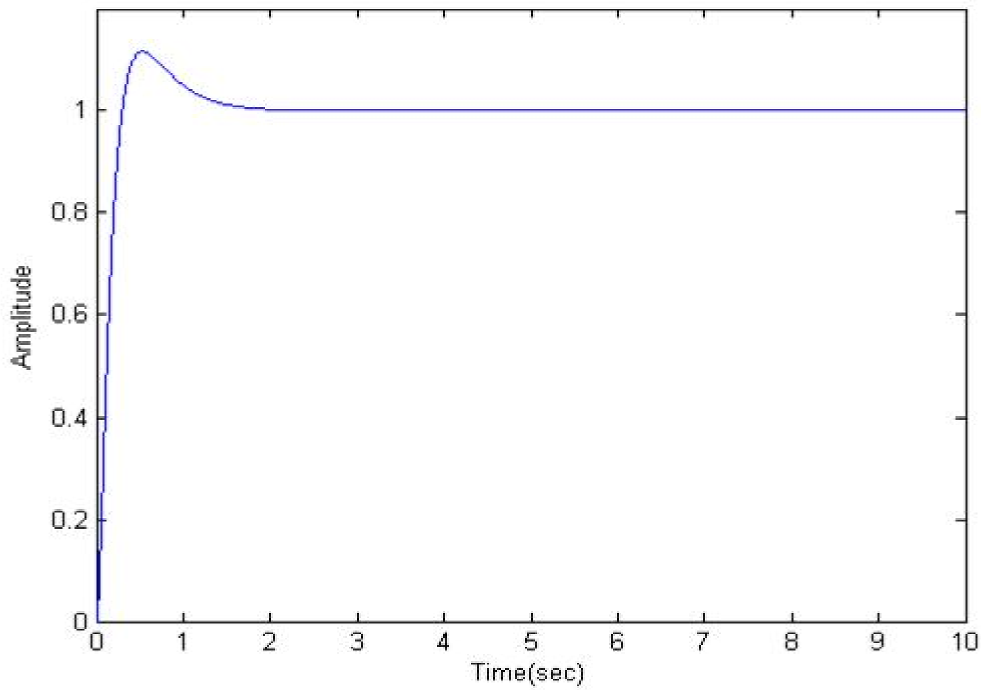

Figure 4.2:Motor speed response with PID controller

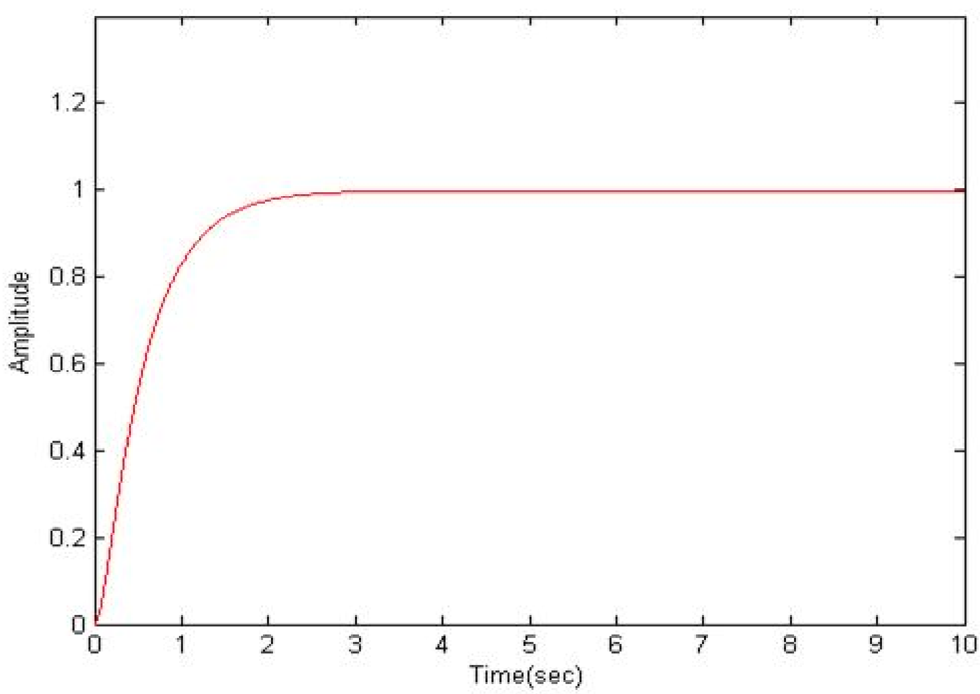

Figure 4.3: Motor speed response with fuzzy controller

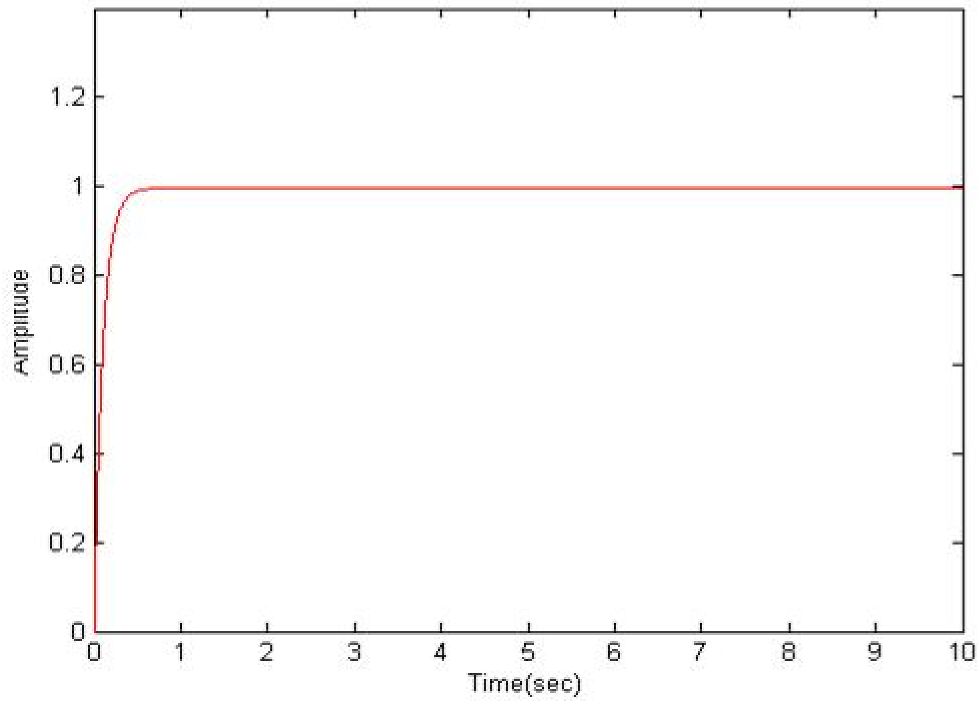

Figure 4.4: Motor speed response with neuro fuzzy controller

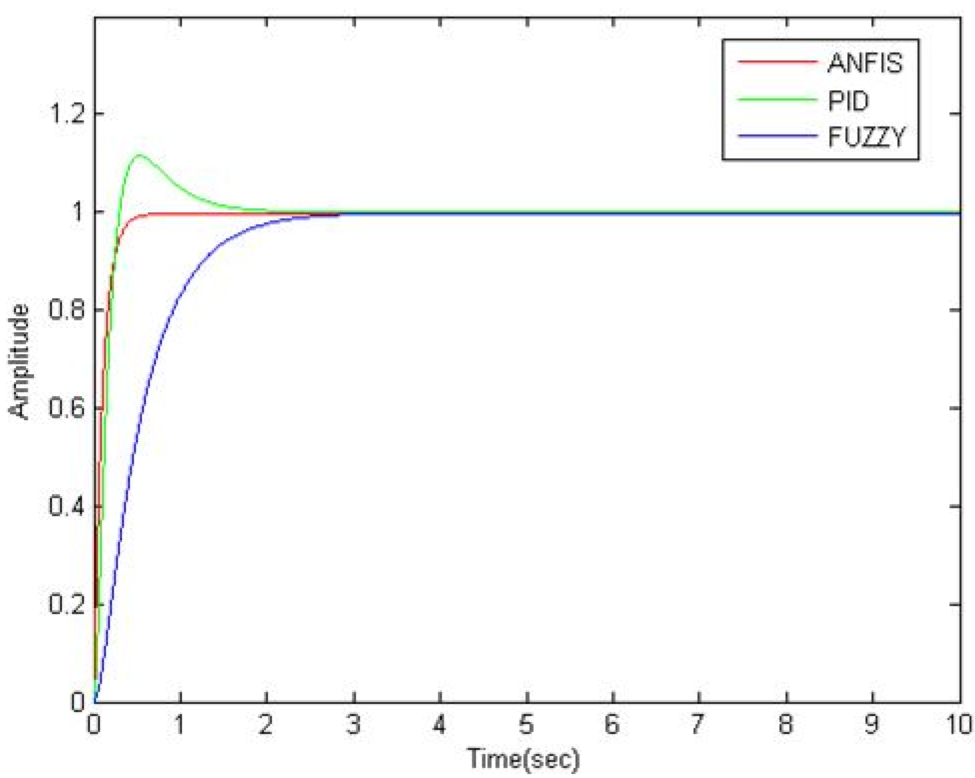

Figure 4.5: Motor speed response with three different controllers

CHAPTER FIVE

CONCLUSION AND RECOMMENDATIONS

CHAPTER FIVE

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

This study is intended to demonstrate the successful application of adaptive neuro-fuzzy controller, fuzzy logic controller and PID controllers for DC motor speed control. The MATLAB/SIMULINK model of the system under study with all three controllers is shown in Figures 3.3-3.6. The performances of these controllers are compared. Simulation results are presented and analyzed for all the controllers. It is observed that fuzzy logic based controller's gives better responses than the traditional and the neuro-fuzzy controller give the best response.

In the PID controller design a lots of attempts are needed to choose the right term, which gives a good response. Design with fuzzy controller gives perfect results, but also a trial-and-error method is needed to find the required parameters. Design with neuro-fuzzy controller reached a very good response and was very fast. The advantages of the neuro-fuzzy controller are that it determines the number of rules automatically, reduces computational time, learns faster and produces lower errors than other methods. With proper design a neuro-fuzzy controllers can replace PID and fuzzy controllers for the speed control of dc motor drives. From simulations, it is concluded that the use of ANFIS reduces design efforts and gives better results.

5.2 Recommendations

- MATLAB simulation for speed control of PMDC motor has been done which can be implemented in hardware to observe actual feasibility of the approach applied in this thesis.
- This technique can be extended to other types of motors.

REFERENCES

REFERENCES

[1] Manafeddin Namzov & Onur Basturk, “DC motor position control using fuzzy proportional-derivative controllers with different defuzzification method”,. An Official Journal of Turkish Fuzzy System association, Vol.1 No.1, pp. 36-54, 2010.

[2] Nader Jamali Sufi Amlashi, “Design and Implementation of Fuzzy Position Control System for Tracking Applications and Performance Comparison with Conventional PID”, IAES International Journal of Artificial Intelligence (IJ-AI),Vol. 1, No. 1, pp. 31 - 44, March 2012.

[3] Hassan Baghgar Bostan Abad, Ali Yazdian Varjani, Taheri Asghar, “Using Fuzzy Controller in Induction Motor Speed Control withConstant Flux”, Proc. of World Aacademy of Science, Engineering and Technology, Vol. 5, April 2005.

[4] Mir. S.A and Malik. E. Elbuluk, “Fuzzy controller for Inverter fed Induction Machines”, IEEE Transactions on Industry Applications, Vol. 30, pp. 78-84, 1994.

[5] Farzan Rashidi, “Sensorless Speed Control of Induction Motor Derives Using a Robust and Adaptive Neuro-Fuzzy Based Intelligent Controller”, IEEE International Conference an Industrial Technology (ICIT), pp. 617-627, 2004.

[6]. Raghavan S, “Digital control for speed and position of a DC motor”, MSs Thesis, Texas A and M University, Kingsville, 2005.

[7] Jinghua Zhong, "PID Controller Tuning: A Short Tutorial", Mechanical Engineering, Purdue University, Spring, 2006.

[8] A.Taifour Ali, Eisa Bashier M. Tayeb and Omar Busati alzain Mohd, "Adaptive PID Controller for Dc Motor Speed Control" International Journal of Engineering Inventions, Volume 1, Issue 5, pp: 26-30, September 2012.

[9] Jeanniton, Christopher James, "Design and development of intelligent navigation control systems for autonomous robots that uses neural networks and fuzzy logic techniques and FPGA for its implementation", Electronic Theses & Dissertations, Paper 770, 2010.

[10] Hunt and Sbarbaro, "Neural Networks for Control System-A Survey", Automatica, Vol. 28, pp. 1083-1112, 1992.

[11] Hung T. Nguyen - Nadipuram R. Prasad - Carol L. Walker - Elbert A. Walker "A First course in Fuzzy and Neural Control" CHAPMAN and HALL/CRC, 2003.

[12] H. X. Li and S. K. Tso, "Quantitative design and analysis of Fuzzy Proportional-Integral-Derivative Control- a Step towards Auto tuning", International journal of system science, Vol. 31, No. 5, pp. 545-553, 2000.

[13] Thana Pattaradej, Guanrong Chen and PitikhateSooraksa, "Design and Implementation of Fuzzy PID Control of a bicycle robot", Integrated computer-aided engineering, Vol.9, No.4, 2002.

[14] Weiming Tang, Guanrong Chen and Rongde Lu, "A Modified Fuzzy PI Controller for a Flexible-joint Robot Arm with Uncertainties", Fuzzy Set and System, 118, pp 109-119, 2001.

[15] N. Govind, and A. R. Hasan, "Real Time Fuzzy Logic Speed Control Using Conventional, Assembly and Simulation Methods for Industrial DC Motors, IEEE/IAS International Conference on Industrial Automation and Control, pp. 203-208, 1995.

[16]NurAzliza Ali, “Fuzzy logic controller for controlling DC motor speedusing MATLAB applications”, Faculty of Electrical and Electronic Engineering, University Malaysia Pahang, November 2008.

[17] J. R. Jang, “ANFIS: Adaptive-Network-Based Fuzzy Inference System”, IEEE Trans. On Systems, Man and Cybernetics, vol. 23, No. 3, pp 665-685, May 1993.

[18] A.Taifour Ali, Eisa Bashier, Mirghani Fateh Elrahman and Ghazally Y. Mustafa, “Neuro-Fuzzy Controller Design for a DC Motor Drive”, University of Khartoum Engineering Journal (UofKEJ), Vol. 3, Issue 1, pp. 7-11, February 2013.