Implementation of a Neuro-Fuzzy PD Controller for Position Control

Sudipta Chakraborty
Department of Electronics Engineering
Indian School Mines Dhanband

Saunak Bhattacharya
Department of Electronics Engineering
Techno India Chaibasa

Debabrata Raha
Department of Electronics Engineering
Techno India Chaibasa

Abstract- Neural network simulations appear to be a recent development. However, this field was established before the advent of computers, and has survived at least one major setback and several eras. The history of neural networks can be traced back to the work of trying to model the neuron. Neural networks have the natural ability of learning like that of a human brain. This paper focuses on the implementation of Neural network based fuzzy PD controller for position control of a DC motor. Firstly, we have used the fuzzy logic controller of Mamdani type inferencing. The training data is fed to the Neural network based structure (connectionist structure or ANFIS structure). That training data is considered as the expert data. The proposed ANFIS structure uses the Sugeno type inferencing. In both of the fuzzy and Neuro-fuzzy PD controller has the rule base. It is verified that the DC motor characteristics is better using the ANFIS controller with lesser number of rules than that of the fuzzy logic controller.

Keyword: ANFIS controller, connectionist model, fuzzy PD controller, neuro-fuzzy PD controller, DC motor position control.

I. INTRODUCTION

During the past few years, Fuzzy systems and neural networks have attracted the interest of researchers in various scientific and engineering areas [1], [2] and variety of applications of fuzzy logic [3], and neural networks [4], [5], ranging from consumer electronics [6], [7] and industrial process control[8], [9] to decision analysis [10], medical instrumentation [11] and financial trading. The happy marriage of the techniques of fuzzy logic and neural networks suggests the novel idea of transforming the burden of designing fuzzy logic [12] systems to the training and learning of connectionist neural network and vice-versa Neuro-fuzzy modeling [13], together with a new driving force from stochastic, gradient-free optimization techniques such as genetic algorithms and simulated annealing, forms the constituents of so–called soft computing [13], which is aimed at solving real world decision–making modeling, and control problems. These problems are usually imprecisely defined and require human intervention. Thus neuro-fuzzy and soft computing, with their ability to incorporate human knowledge and to adapt their knowledge base via new optimization techniques, are likely to play increasingly important roles in the conception and design of hybrid intelligent systems. These benefits can be witnessed by the success in applying neuro-fuzzy systems in various areas.

The combination of neural networks and fuzzy logic offers the possibility of modification of fuzzy membership functions and design difficulties of fuzzy logic [14]. This new approach combines the well established advantages of both the methods and avoids the drawbacks of both. The resultant network will be more transparent and can be easily recognized in the form of fuzzy logic control rules.

An Artificial Neural Network (ANN) [4], [5] is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information. Artificial Neural Networks, also referred as to connectionist systems or neurocomputing, are the recent generations of information processing systems that are constructed to make use of some organizational principles that characterize the human brain. Artificial neural networks may either be used to gain an understanding of biological neural networks, or for solving artificial intelligence [15] problems without necessarily creating a model of a real biological system. An

ANN is configured for a specific application, such as pattern recognition [16] or data classification, through a learning process [17]. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurons.

In modern software implementations of artificial neural networks [4], [5], the approach inspired by biology has been largely abandoned for a more practical approach based on statistics and signal processing. In some of these systems, neural networks or parts of neural networks (such as artificial neurons) are used as components in larger systems that combine both adaptive and non-adaptive [18] elements. While the more general approach of such adaptive systems is more suitable for real-world problem solving, it has far less to do with the traditional artificial intelligence connectionist models. Historically, the use of neural networks models marked a paradigm shift in the late eighties from high-level (symbolic) artificial intelligence [15], characterized by expert systems with knowledge embodied in if-then rules, to low-level (sub-symbolic) machine learning, characterized by knowledge embodied in the parameters of a dynamical system. The next major development in neural networks came in 1949 with the publication of Hebb's book *The Organization of Behavior*, in which an explicit statement of a physiological learning rule for *synaptic modification* [19] was present for the first time. He introduced his famous *postulates of learning*, which states that the effectiveness of a synapse between two neurons is increased by the repeated activation of one neuron by the other across that synapse.

Later, as computers emerged in the 1950s, several researchers attempted to utilize the new technology to create better neural networks [20]. Over the next decade or so these physiologists, psychologists and computer engineers contributed greatly to the development of artificial neural networks.

Significant progress has been made in the field of neural networks-enough to attract a great deal of attention and fund further research. Advancement beyond current commercial applications appears to be possible, and research is advancing the field on many fronts. Clearly, today is a period of transition for neural network technology. Although the golden age of neural network research ended 25 years ago, the discovery of back propagation has reenergized the research being done in this area. In recent times, networks with the same architecture as the back propagation network are referred to as multilayer perceptrons. This name does not impose any limitations on the type of algorithm used for learning. The back propagation network generated much enthusiasm at the time and there was much controversy about whether such learning could be implemented in the brain or not, partly because a mechanism for reverse signaling was not obvious at the time, but most importantly because there was no plausible source for the 'teaching' or 'target' signal. However, since 2006, several unsupervised learning procedures [21] have been proposed for neural networks with one or more layers, using so-called deep learning algorithms. These algorithms can be used to learn intermediate representations, with or without a target signal, that capture the salient features of the distribution of sensory signals arriving at each layer of the neural network.

II. GENERAL FUZZY LOGIC CONTROL AND DECISION SYSTEM

(a) General Fuzzy Logic Structure

This section introduces the structure and functions of our previously proposed Neural-Network-Based Fuzzy Logic Controller (NN-FLC) [22], [23], which is a basic component of the proposed RNN-FLCS. The learned NN-FLC functions as a connectionist neural-network-based fuzzy logic control and decision-making system.

Fig.1 shows the basic configuration of a fuzzy logic controller which is composed of three major components: fuzzifier, fuzzy rule base and inference engine, and defuzzifier. The fuzzifier performs the function of fuzzification that converts input data from an observed input space into proper linguistic values of fuzzy sets through predefined input membership functions. The rule base consists of a set of fuzzy logic rules in the form of "IF & THEN" to describe the control policy of expert knowledge. The inference engine is to match the output of the fuzzifier with the fuzzy logic rules and perform fuzzy implication and approximate reasoning to decide a fuzzy control action. Finally, the defuzzifier performs the function of defuzzification to yield a nonfuzzy (crisp) control action from an inferred fuzzy control action through predefined output membership functions.

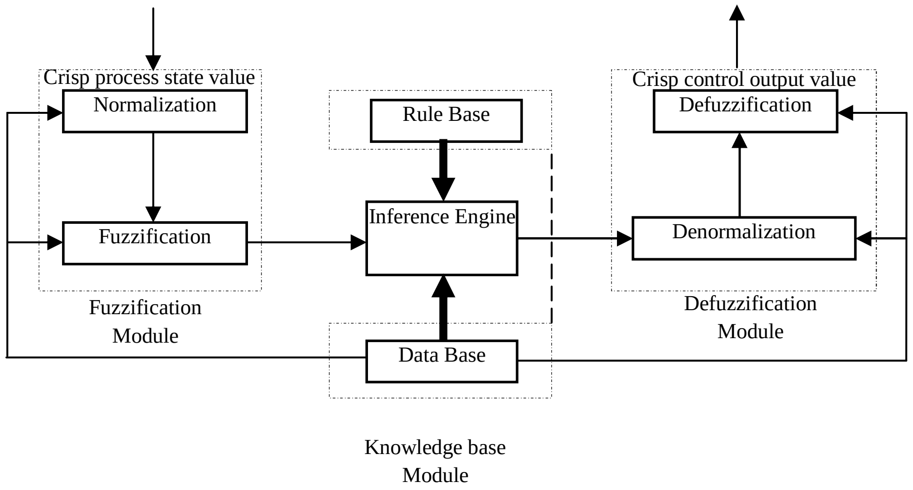

Figure 1. Block diagram of FKBC.

In fuzzy logic systems the decisions are based on the inputs. Those inputs are in the form of linguistic variables derived from the membership function that is used to determine the fuzzy set. The linguistic variables are then matched with the preconditions of linguistic IF-THEN rules and the result of each rule is obtained through fuzzy implication. These 'IF-THEN' rules are stored in rule base. The rule base for our work is shown in fig.2. Now the response of each rule is weighted according to the degree of membership of its inputs and the centroid of the response is calculated to generate the appropriate decision output. In that way we can perform compositional rule of inferencing. The most accepted approach is to define the rules and the membership functions from a human operated system or an existing controller and testing the design for proper output.

Δe / e	NB	NS	ZE	PS	PB
NB	**NB**	**NB**	**NB**	**NS**	**ZE**
NS	**NB**	**NB**	**NS**	**ZE**	**PS**
ZE	**NB**	**NS**	**ZE**	**PS**	**PB**
PS	**NS**	**ZE**	**PS**	**PB**	**PB**
PB	**ZE**	**PS**	**PB**	**PB**	**PB**

Figure 2. Fuzzy rule base for 25 rules.

The rule base is same as the Fuzzy controller rule base. Now we shall embed the features of the ANN into the Fuzzy logic structure. This hybrid model is called *Connectionist fuzzy logic model*. We shall describe the detailed functions of the nodes in each of the five layers of the proposed connectionist model. The input and output linguistic notations are presented by the nodes in the input layer and output layer respectively in the connectionist fuzzy logic system.

(b)Connectionist structure

This section introduces the structure and functions of the proposed neural fuzzy control network (*NFCN*), which is a feed forward multilayered connectionist structure. The NFCN integrates the basic elements and functions of a traditional FLC (e.g., membership functions, fuzzy logic rules, fuzzification, defuzzification, and fuzzy implication) into a connectionist structure which has distributed learning abilities to learn the input/output membership functions and fuzzy logic rules. Fig. 3 shows the structure of our proposed NFCN. Nodes at *layer one* are input nodes (linguistic nodes) that represent input linguistic variables. *Layer five* is the output layer. The five layered connectionist structure performs fuzzy inference. Nodes at *layer one* are input nodes whose inputs are fuzzy numbers or crisp number. Each input node corresponds to one input linguistic variable. Nodes at *layer two* and *four* are the term nodes. It acts as membership functions to represent the terms of the respective linguistic variable. Actually, a *layer-two* node can be either a single node or composed of multilayer nodes. Nodes *Layer five* consists of output nodes whose outputs are also fuzzy number or crisp numbers. Only *layer two* and *layer five* have fuzzy weight. The single node performs a simple membership function. The simple membership function can be a triangular or bell shaped function. The multilayer nodes perform a complex membership function. A sub-neural network is an example of multilayer nodes. Each node in *layer two* executes a match action to find the match degree between the input fuzzy number and the fuzzy weight. The nodes at *layer three* are rule nodes. Each node represents one fuzzy rule. Thus, we can summarize that all the nodes at this layer together forms a fuzzy rule base. The links at *layer three* and *four* function as a connectionist inference engine that avoids the rule matching process. Layer *three* links define the preconditions of the rule nodes and layer four links define the consequences of the rule nodes. The links at layer *two* and *five* are fully connected between linguistic nodes and their corresponding term nodes. Now the network will learn fuzzy logic rules by deciding the existence and connectionist types of the links of layer three (precondition links) and layer four (consequence nodes).

With this five-layered structure of the proposed connectionist model, the basic functions of a node can be defined. A typical network consists of a unit which has some finite fan-in of connections represented by weight values from other units and fan-out of connections to other units. Associated with the fan-in of a unit is an integration function f which serves to combine information, activation, or evidence from other nodes.

The applications of the back-propagation algorithm two distinct passes of computation are distinguished. The first one is the *Forward pass* and the second one is *Backward pass*. In the forward pass the synaptic weights remain unaltered throughout the network and the function signals in the network are computed on a neuron-by-neuron basis. And in the backward pass the synaptic weights of the neurons are adjusted by the process of Back-propagation learning algorithm.

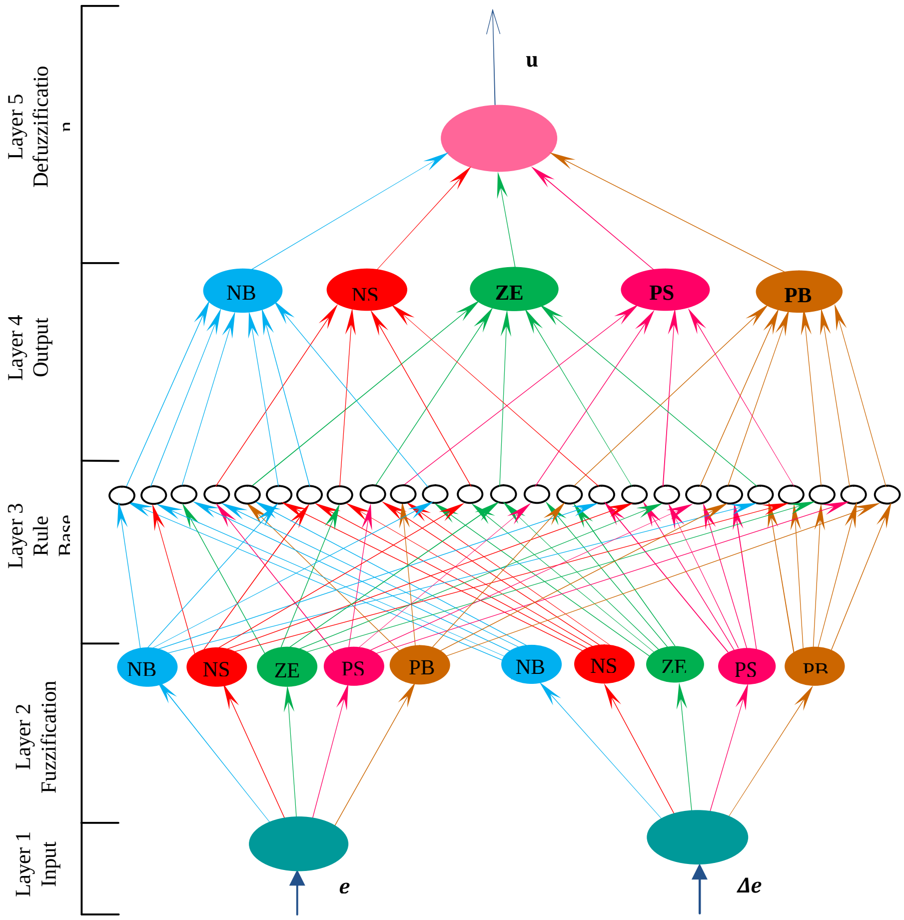

Figure 3. Connectionist fuzzy logic control/ decision system for the forward pass of the neural network.

Forward Pass

Layer 1 :

The nodes in this layer just transmit input values to the next layer directly *i.e.* this layer is transparent.

$$f \qquad u_i$$
$$\text{and } a = f \tag{1}$$

The link weight at layer one ($_i$) is unity.

Layer 2 :

Let us consider a single node that is to perform a simple membership function. We take the "bell shaped" function. For example –

$$f \qquad ^{j} \qquad _{ij}\,,\ _{ij} \qquad - \frac{u_i - \quad _{ij}}{_{ij}}$$
$$\text{And, } a \quad e \tag{2}$$

Here $_{ij}$ = Centre (or mean) of the bell shaped function of the term of the i input linguistic variable $_i$ and, $_{ij}$ = Width (or variance) of the bell shaped function of the term of the i input linguistic variable $_i$. Hence the link weight at layer two $_{ij}$ can be interpreted as $_{ij}$. If we use a set of nodes to perform a membership function, then the function of each node can be just in the standard form [8]. The whole subnet is learned off-line by a learning algorithm (*e.g.* Back-propagation) to perform the desired membership function.

Layer 3 :

The links in this layer are used to perform preconditions matching of fuzzy logic rules. Hence, the rule nodes should perform the fuzzy AND operation.

$$f = min\ u\ {}_{,}u\ , \ldots \ldots ..,u \qquad \text{and} \qquad a = f \tag{3}$$

i.e. layer 3 is transparent layer and its weight ($_i$) is then unity.

Layer 4 :

The nodes in this layer have two operational modes –

i) Down-up transmission mode.
ii) Up-down transmission mode.

In down-up transmission mode, the links of layer 4 should perform the fuzzy OR operation to integrate the fired rules that have the same consequences.

$$f \qquad u_i$$
$$i \tag{4}$$

And, $a = min\ (1, f\ )$

The link weight is then $_i$ $= 1$. In up-down transmission mode, the nodes of layer 4 and links of layer 5 functions same as layer 2.

Layer 5 :

There are two kinds of nodes in this layer. The first kind of node performs the down-up transmission for the decision signal output.

$$f = \quad {}_i \text{ and } a = f \tag{5}$$

The second kind of node performs the down up transmission mode for the decision signal output.
If $_{ij}$ and $_{ij}$ are the centre and the width of the membership function respectively, then

$$f \quad \Sigma \quad {}_{ij}u_i \quad \Sigma \quad {}_{ij} \quad {}_{ij} \quad u_i$$
$$\text{And, } a \quad \frac{\quad}{\Sigma} \tag{6}$$

These functions can be used to simulate the 'Centre of Area' defuzzfication method.

Supervised Learning Phase

Supervised training is the process of providing the network with a series of simple input and comparing the output with the expected responses. The training continues until the network is able to provide the expected response. This learning process is considered as learning with a teacher and it is thought that teacher has knowledge of the environment. This knowledge is represented by a set of input-output. In the supervised learning method the actual error is being minimized to obtain the actual control action output. The idea of Back propagation is used in this learning phase. During the Backward pass the synaptic weights are all adjusted in accordance with an 'error correction rule'.

Layer 5 :

In this layer the centre and width parameter of the membership function is updated using Back-propagation learning algorithm and those updated parameters are denoted as $_i\ t \quad 1$ and $_i\ t \quad 1$ respectively. As in the layer 5, there is same error signal is propagated by taking the difference between the desired output and current output signal and these error signal goes to the output all term nodes in layer 4.

$$E \quad \frac{1}{2} \quad e \quad \frac{1}{2} \quad t - \quad t \tag{7}$$

As in forward learning the result will be terminated from 5th layer. So in back propagation learning this result's adjustment will be started from 5th layer to minimize the error function (E). The weight can be interpreted as the centre $_i$ and width $_i$. To adjust the centre $_i$ and width $_i$ is adjusted in the layer five by using the chain rule $\frac{E}{\quad}$ will be,

$$\frac{\;}{_i} \quad \frac{\;}{}\frac{\;}{}\frac{\;}{_i} \quad \frac{\;}{}\frac{\;}{}\frac{\;}{}\frac{\;}{_i} \qquad (8)$$

$$\frac{\;}{} \quad - \frac{1}{2} e \quad e \quad t - \quad t \qquad (9)$$

$$\frac{\;}{} \quad \frac{\;}{t} \quad \frac{\;}{t} \quad t - \quad t \quad -1 \qquad (10)$$

$$\frac{\;}{} \quad \frac{\;}{}\frac{f}{\sum \;_{ij} u_i} \quad \frac{1}{\sum \;_i u_i} \qquad (11)$$

$$\frac{\;}{_i} \quad \frac{\;}{_i} \quad _{ij} \;_{ij} \; u_i \quad _i u_i \qquad (12)$$

We define the instantaneous value of the error energy for a particular neuron as $-e$ .Correspondingly the instantaneous value of E of the total error energy is obtained by summing $-e$ over all neurons in the output layer. The instantaneous error energy is a function of the free parameters (the centre and width of the membership function).

$$\frac{\;}{_i} \quad - \quad t - \quad t \; \frac{u_i \;_i}{\sum u_i \;_i} \qquad (13)$$

So the centre parameter value will be updated as follows,

$$\begin{aligned} &_i t \quad 1 \quad _i t \quad - \frac{\;}{_i} \quad _i t - \quad - \quad t - \quad t \; \frac{u_i \;_i}{\sum u_i \;_i} \\ &_i t \quad t - \quad t \; \frac{\;}{\sum} \end{aligned} \qquad (14)$$

Next we have to adjust the width parameter so,

$$\frac{\;}{_i} \quad \frac{\;}{}\frac{\;}{}\frac{\;}{} \quad \frac{\;}{}\frac{\;}{}\frac{\;}{}\frac{\;}{_i} \qquad (15)$$

$$\begin{aligned} &\frac{\;}{} \quad - \quad t - \quad t \\ &- \; 2 \quad t - \quad t \quad -1 \end{aligned} \qquad (16)$$

$$a \quad \frac{f}{\sum u_i \;_i} \; and, \quad f \quad \frac{_i \;_i u_i}{\sum \;_i u_i}$$

So by putting the value of 'f'

$$\begin{aligned} &a \quad \frac{_i \;_i u_i}{\sum \;_i u_i} \\ &\frac{\;}{_i} \quad \frac{\sum u_i \;_i \; u_i \;_i - \sum \;_i \;_i u_i \; u_i}{\sum \;_i u_i} \end{aligned} \qquad (17)$$

$$\frac{\;}{_i} \quad \frac{\;}{}\frac{\;}{_i} \quad - \quad t - \quad t \quad \frac{\sum \;_i u_i \quad _i u_i - \sum \;_i \;_i u_i \; u_i}{\sum \;_i u_i} \qquad (18)$$

So the width parameter will be updated as follows,

$$_i t \quad 1 \quad _i t \quad - \frac{\;}{_i} \quad _i t \quad t - \quad t \quad \frac{\sum \;_i u_i \quad _i u_i - \sum \;_i \;_i u_i \; u_i}{\sum \;_i u_i} \qquad (19)$$

So the error to be propagated to the preceding layer is,

$$-\frac{i}{f} \quad - - \quad t - \quad t \qquad t - \quad t \tag{20}$$

Layer 4 :

In this learning the adjustment calculation is done in layer5. Only the updated value from layer 5 is taken to adjust synaptic weight (in terms of centre and width parameter) to the layer 4. In forward down up mode, no parameter is needed to be adjusted. Only error signals ($_i$) needed to computed and propagated. So the error signal $_i$ is needed to be derived in the following way. So taking chain rule,

$$- \ _i \quad \frac{i}{} \quad \frac{i}{a_i}\frac{a_i}{f_i} \quad \frac{i}{net\ input}\frac{net\ input}{a_i} \tag{21}$$

After passing through the activation function if net input is equal to the result of activation function then the content of '*f*' is just like '*a*'. So,

$$\frac{net\ input}{a_i} \quad \frac{f}{u_i} \quad \frac{a}{u_i} \tag{22}$$

$$\text{As}, a \quad \frac{f}{\sum {}_i u_i} \quad \frac{\sum {}_i {}_i u_i}{\sum {}_i u_i}$$

$$\frac{}{u_i} \quad \frac{\sum {}_i u_i \quad {}_i {}_i - \sum {}_i {}_i u_i \quad {}_i}{\sum {}_i u_i} \tag{23}$$

$$\frac{i}{net\ input} \quad \frac{i}{f} \quad - \quad - \quad t - \quad t \tag{24}$$

So, the error signal will be,

$$_i \ t \quad - \quad t - \quad t \quad \frac{{}_i {}_i \sum {}_i u_i - \sum {}_i {}_i u_i \ {}_i}{\sum {}_i u_i} \tag{25}$$

Layer 3 :

In *layer 4* the error signal is need to be computed. This error signal can be derived as

$$_i \quad - \frac{i}{} \quad - \quad \frac{i}{\text{net input}}\frac{net\ input}{a_i} \quad - - \ {}_i \quad {}_i \tag{26}$$

So the error signal is, $_i$ = $_i$.

Layer 2 :

In layer 2, the centre and width parameter of the membership function is denoted as $_{ij}$ and $_{ij}$ respectively. In this layer *i* and *j* term is associated with m (centre) and (width) to indicate the centre and width of bell shaped function of the j^{th} term of the i^{th} input linguistic variable $_i$. Calculation is done but calculated value is applied to this layer to adjust the weight. The adaptive rule of $_i$ is derived as following way by using the chain rule,

$$\frac{i}{{}_{ij}} \quad \frac{i}{a_i}\frac{a_i}{f_i}\frac{f_i}{{}_{ij}} \tag{27}$$

$$\frac{i}{a_i} \quad \frac{i}{net\ input}\frac{net\ input}{a_i} \tag{28}$$

$$\frac{i}{net\ input} \quad \frac{i}{f} \tag{29}$$

And,

$$\frac{i}{} \quad \frac{}{} \quad \begin{cases} 1, \text{ If } u_i = \text{min (input of rule node k)} \\ 0 \text{ for other} \end{cases} \tag{30}$$

The output will be '0' if partial derivative is done with other parameter.

$$\frac{i}{a_i} \tag{31}$$

and

If a_i is minimum in k^{th} rule node's input

otherwise $= 0$ (33)

$a \quad e$ so, $\frac{a_i}{} \quad e$ (32)

And $f \quad - \frac{}{}$

so, $\frac{f_i}{ij} \quad - \quad 2\frac{u_i - \quad ij \quad -1}{ij} \quad \frac{2 \; u_i - \quad i}{ij}$ (33)

So the updated value of $_i$ can be calculated as follows and the adaptive rule of $_i$ is,

$$_{ij}\; t \quad 1 \qquad _{ij}\; t \qquad - \frac{i}{ij} \qquad _{ij}\; t \qquad \frac{i}{a_i} e \; \frac{2 \; u_i - \quad ij}{ij} \tag{34}$$

The adaptive rule of $_{ij}$ is derived as,

$$\frac{i}{ij} \quad \frac{i}{a_i}\frac{a_i}{f_i}\frac{f_i}{ij} \tag{35}$$

$\frac{E}{}$ and $\frac{}{}$ values are calculated in the above equation and those value will be put into this equation,

$$\text{Now,} f \quad - \quad \frac{u_i \; - \quad ij \quad ij}{1}$$

$$\frac{f_i}{ij} \quad - \; -2 \; _{ij} \qquad u_i - \quad _{ij} \qquad \frac{2 \; u_i - \quad ij}{ij} \tag{36}$$

So the updated value of $_{ij}$ is derived as,

$$_{ij}\; t \quad 1 \qquad _{ij}\; t \qquad - \frac{i}{ij} \tag{37}$$

It should be also noted that this backpropagation algorithm can be easily extended to train the membership function which is implemented by a subneural network instead of a single term node at layer two, since, from the above analysis, the error signal can be propagated to the output node of the subneural network. Then, by using a similar backpropagation rule in the subneural net, the parameters in the subneural network are adjusted.

III. EXPERIMENTAL SET-UP AND RESULTS

In the fuzzification process, i.e., in the first stage, the crisp variables, the speed error & the change in error are converted into fuzzy variables or the linguistics variables. The fuzzification maps the two input variables to linguistic labels of the fuzzy sets. The fuzzy rules are fired by Sugeno type inferencing technique. The fuzzy controller uses the linguistic labels. Each fuzzy label has an associated membership function. Those rules are fired gets the output to the defuzzification module. This module defuzzifies the output to the crisp value because every real world problems are dealt with the crisp logic. We have used the Sugeno type inferencing technique because it is a more compact and computationally efficient representation than a Mamdani system, the Sugeno system lends itself to the use of adaptive techniques for constructing fuzzy models. These adaptive techniques can be used to customize the membership functions so that the fuzzy system best models the data.

The main advantages of Sugeno type inferencing are:

- It is computationally efficient.
- It works well with optimization and adaptive techniques.
- It has guaranteed continuity of the output surface.
- It is well suited to mathematical analysis.

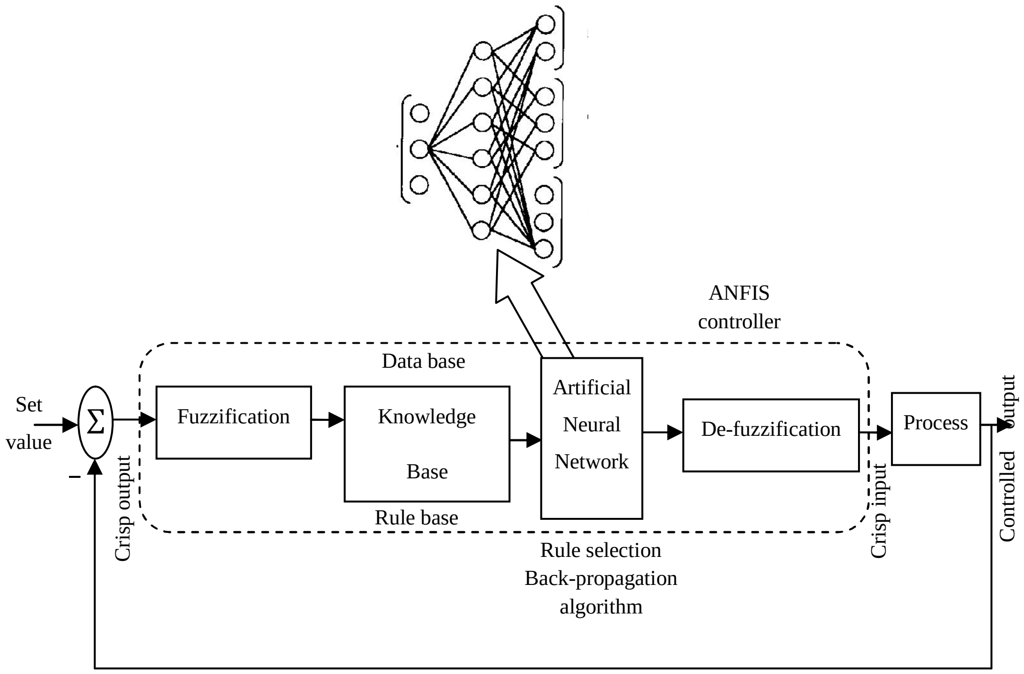

Figure 4. Block diagram of neuro-fuzzy controller.

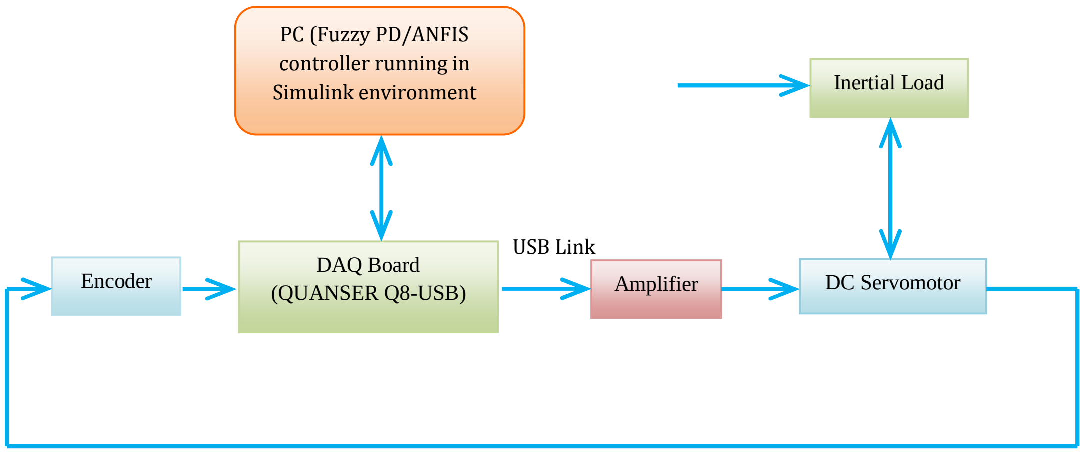

Figure 5. Block diagram of the actual closed loop system using neuro-fuzzy controller.

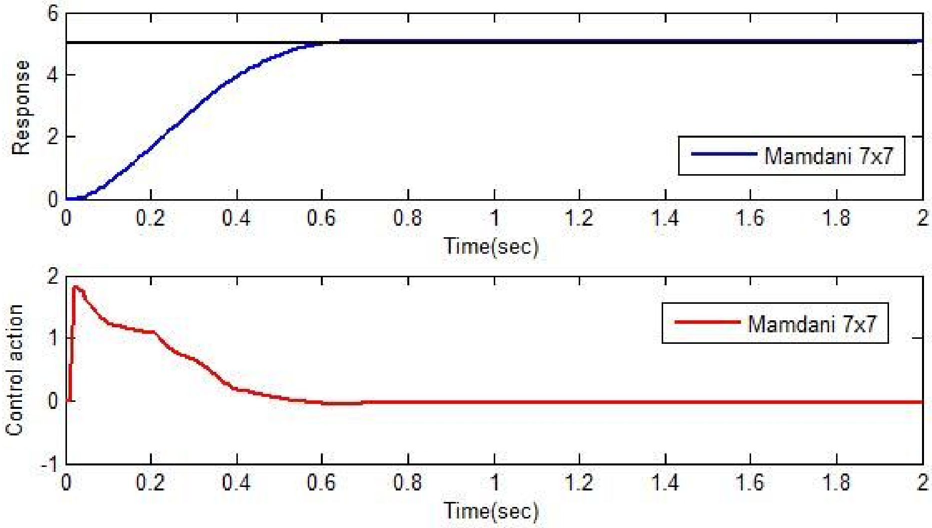

Figure 6. Closed loop response and control action of DC motor using Mamdani FLC (49 rules).

At first we have set the Mamdani type FLC with the DC Servomotor in closed loop. The closed loop response and the control action taken by the Fuzzy PD controller (rule base contain 49 rules) is shown if fig.(6)

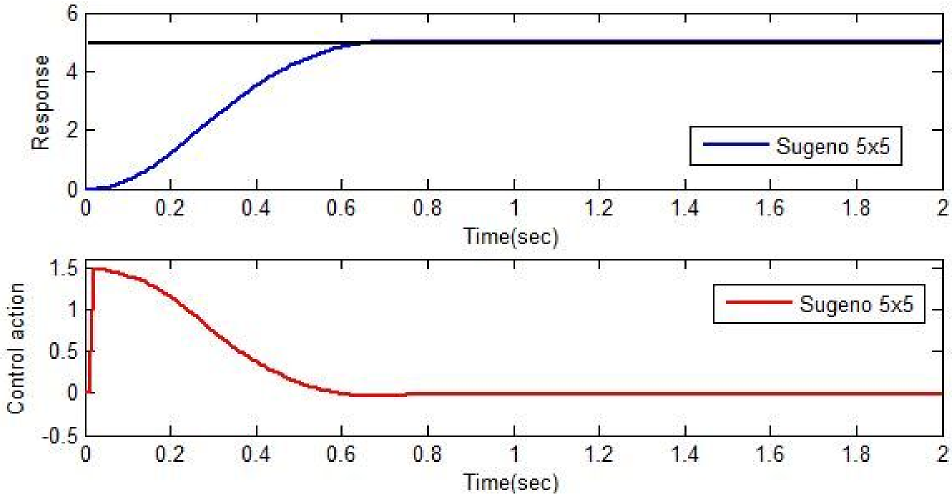

Figure 7. Closed loop response and control action of DC motor using the ANFIS(25 rules).

Now we are substituting the Fuzzy PD controller by the ANFIS controller (25 rules) and observe the response and the control action taken by the ANFIS controller. They are shown in the fig.(7).

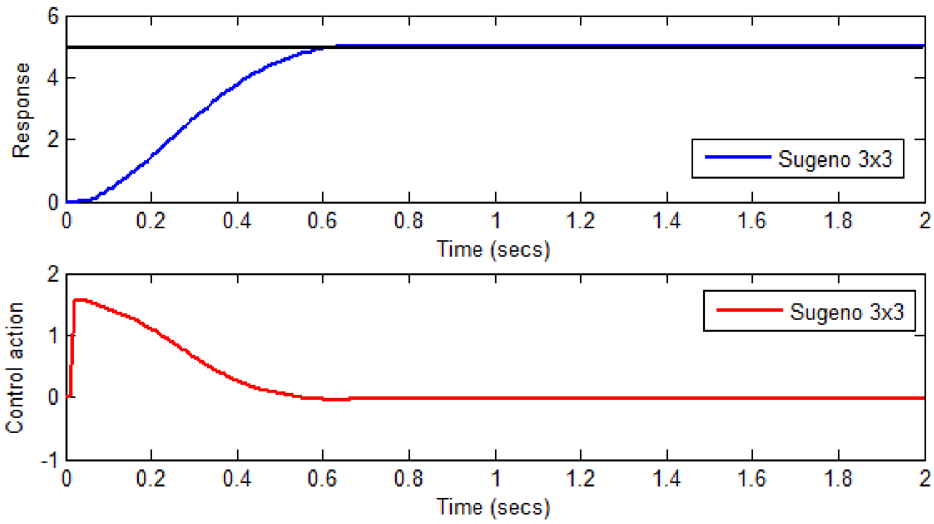

Figure 8. Closed loop response and control action of DC motor using ANFIS (9 rules).

Finally we use the ANFIS controller (rule base containing 9 rules) and observe the response as shown in fig.(8). We can conclude from the response and the control action curve that by using the Sugeno type inferencing scheme gives better response than the Mamdani type inferencing scheme. It is to be noted that the sugeno type contains lesser number of rules (9 rules) than that of the mamdani type (49 rules). So we can say that the sugeno type inferencing scheme gives better response than the mamdani type inferencing

IV. CONCLUSION

The Fuzzy PD and ANFIS controller designed in this project using MATLAB shows acceptable performance for controlling the Servo position control system in real-time environment. However, the ANFIS controller shows almost same performance as the Fuzzy PD controller for Real-Time performance. Thus we can achieve the same performance by reducing the number of rules without degrading the performance, in case of ANFIS controller. Effect of various design parameters on the performance of the Fuzzy PD controller for controlling the Servo Position Control System and also the robustness of the design to withstand parameter variations is tested here.

The ANFIS structure can further be tested on other processes to establish its robustness to system module variation. If the nature of the ANFIS structure is modified then we do not have to provide the training data to learn the ANFIS. Online tuning strategies can also be implemented on the rule base to modify the rules making it a self-organizing Fuzzy controller. Various optimization techniques and learning algorithms can be used with the fuzzy controller to select the optimum settings of the different design parameters for a particular process.

REFERENCES

[1] K. Asakawa and H. Takagi, "Neural Networks in Japan," *Communication of the ACM*, Vol. 37, no. 3,1994, pp. 106-112.

[2] C. von Altrovk, B. Krause, and H.J. Zimmerman, "Advanced fuzzy logic control technologies in automotive applications," Proc. *IEEE Int. Conf. of Fuzzy Systems*, San Diego, 1992, pp. 835-842.

[3] L.A. Zadeh, "Fuzzy sets," *Infor. Contr.*, vol. 8, pp. 338-353, 1965.

[4] C.T. Lin and C.S.G. Lee, "Neural-network-based fuzzy logic control and decision system," *IEEE Trans. Comput*. Vol. C-40. no. 12, pp. 1320-1336, Dec. 1991.

[5] C.T. Lin and C.S.G. Lee, "Real-time supervised structure/parameter learning for fuzzy neural network," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, San Diego, CA, Mar. 1992, pp. 1283-1290.

[6] S. Shao, "Fuzzy self-organising controller and its application for dynamic processes," Fuzzy Sets and Systems, vol. 26, 1988, pp. 151-164.

[7] H. Takagi, "Application of neural networks and fuzzy logic to consumer products," *Proc. Int. Conf. on Industrial Fuzzy Electronics, Control. Instrumentation, and Automation*, vol. 3, San Diego, Nov. 1992, pp. 1629-1639.

[8] T. Culliere, A. Titli, and J. Corrieu, "Neuro-fuzzy modelling of nonlinear systems for control purposes," *Proc. IEEE Int. Conf. On Fuzzy Systems*, Yokohama, 1995, pp. 641-648.

[9] C.C. Lee, “Fuzzy logic in control system: Fuzzy logic controller- Part I & II,” *IEEE Trans. Syst., Man, Cybern.,* vol. SMC-20, no. 2, pp. 404-435, 1990.

[10] J. Hullatz, “Neuro-fuzzy in legal reasoning,” *Proc. IEEE Int. Conf On Fuzzy Systems,* Yokohama, 1995, pp. 655-662.

[11] T. Chen, “Fuzzy neural network in medicine,” *Proc, IEEE Int. Conf. on Fuzzy Systems,* Yokohama, 1985, pp. 627-634.

[12] D. Nauck, F. Klawonn, and R. Kruise, “Combining neural networks and fuzzy controllers,” *In E. P. Klement and W. Slany, editors, Fuzzy Logic in Artificial Intelligence,* Springer Verlag, Berlinn, 1993, pp. 35-46.

[13] J.S.R. Jang, C.T. Sun, E. Mizutani, “*Neuro-Fuzzy and Soft Computing”* Pearson Education, 1997.

[14] P. J. Werbos, “Neurocontrol and fuzzy logic: connections and design,” *Int . J. Approximate Reasoning,* vol. 6, Feb. 1992, pp. 185-220.

[16] K. Fukushima, S. Miyaka and T. Ito, “Neocognotron: A neural network model for a mechanism of visual pattern recognition,” *IEEE Trans. Syst., Man, Cybern.,* vol. SMC-13, no. 5, pp. 834-847, 1983.

[17] A. Dickinson, *Contemporary Animal Learning Theory.* Cambridge, MA: Cambridge Univ. press, 1980.

[18] F.C. Chen, “Back-propagation neural network for nonlinear self-tuning adaptive control,” Proc. *IEEE Intelligent Machine,*pp. 274-279, 1989.

[19] S. Haykin, “*Neural Networks*” Prentice Hall, 1999.

[20] H. Ishibuchi, R. Fujioka and H. Tanaka, “Neural Networks that learn from fuzzy if-then rules,” *IEEE Trans. Fuzzy Syst.* Vol. 1, no. 2, pp. 85-97, 1993.

[21] G.E. Hinton, “Connectionist learning procedures,” *Artif. Intell.* Vol. 40, no. 1, pp. 143-150, 1989.

[22] C.C.Lee, “fuzzy logic in control systems: Fuzzy logic controller-Part I & II,” *IEEE Trans. Syst.,Man, Cybern., vol.* SMC-20, no. 2, pp.404-435, 1990.

[23] L.A.Zadeh, Fuzzy logic”, *IEEE Comput. Mag.,* pp. 83-93, Apr. 1988.