An Adaptive Neuro-Fuzzy Architecture for Intelligent Control of a Servo System and its Experimental Evaluation

Ayse Cisel Aras, Erdal Kayacan, Yesim Oniz, Okyay Kaynak
Bogazici University
Dept. of Electrical and Electronics Engineering
34342, Bebek, Istanbul, Turkey
Email: {cisel.aras, erdal.kayacan, yesim.oniz, okyay.kaynak}@boun.edu.tr

Rahib Abiyev
Near East University
Dept. of Computer Engineering
Lefkosa, North Cyprus
Email: rahib@neu.edu.tr

Abstract—In this paper the development of an adaptive neuro-fuzzy architecture for the speed control of a servo system with nonlinear load is presented. The synthesis of the structure is described and a learning algorithm for the neuro-fuzzy control system is derived. The supervised learning algorithm is used to train the unknown coefficients of the system, and then the fuzzy rules of the neuro-fuzzy system are generated. A number of simulation studies are carried out, and the results are compared with those obtained with a PI controller tuned using desired time response characteristics. These and the experimental studies presented show that the neuro-fuzzy control system has a better control performance than the conventional PI controller.

I. INTRODUCTION

Increasing complexity of the technological processes and the environmental uncertainties lead to an increasing demand on advanced control structures, as the control algorithms developed on the base of traditional approaches turn out to be rather complex and their implementation might be difficult. In addition, the frequently changing characteristics of the environmental conditions in the form of various disturbance forces necessitate the use of some intelligent approaches with self-training and adapting capability. One such approach is the use of fuzzy neural networks.

The combination of the fuzzy systems and the neural networks has recently become a popular approach in engineering fields to solve control, identification, prediction, pattern recognition, etc. problems [1]. One well known structure is the adaptive neuro-fuzzy inference system (ANFIS) [2]. Others include self-constructing neural fuzzy inference networks [3], TSK-type recurrent fuzzy networks [4] and fuzzy wavelet neural networks [5] that have been used for identification and control purposes. In the literature, fuzzy neural systems (FNSs) are used in a variety of applications, such as for drive systems [6], [7], technological processes [8] and DC converters [9]. In [7] a variable structure system theory based training algorithm is proposed and its use in a neuro-adaptive scheme [10] for the control of electrical drives is described and experimental results are presented. In [11] a robust control system with a fuzzy sliding mode controller coupled with a compensator is presented. The compensator relaying on the sliding mode theory is used to improve the dynamical characteristics of the drive system. A cascade control scheme based on a second-order sliding-mode control algorithm is suggested in [12]. In this paper the development of an adaptive neuro-fuzzy architecture for intelligent control of a servo system with a time-varying load is considered.

Direct Current (DC) motors are often used in industrial control applications where a wide range of speed control is required [13]. DC motors are able to deliver three or more times their rated torque momentarily, and supply over five times rated torque in emergency situations. One of the features of DC motors is that the speed of the motor can be controlled smoothly down to zero. Moreover, DC motors can respond quickly to changes in control signals due to the high ratio of torque to inertia it has [14].

If the motor parameters can be obtained precisely, the control of a DC motor based servo system is a relatively easy problem and a number of model-based approaches, such as PID, pole placement, etc. are widely used. However, in real life, because of noise both from the inside and the outside of the system, the information we can obtain about the system in hand is always uncertain and limited in scope [15]. Furthermore, the load characteristics of the servo systems can be often nonlinear. In such cases, not only does the performance of the model-based approaches dramatically decrease but also the complexity of the controller design increases. The uncertainties are generally coming from the noise in the measurements and the parameter changes due to the environmental and the operating conditions. These have been the motivations behind the use of an adaptive neuro-fuzzy control scheme presented in this study.

Recently, fuzzy logic systems (FLSs) are used for the DC motor control applications [16], [17]. In [18], a neuro-fuzzy network-based control algorithm for the speed control of a DC motor is proposed. In [19], an adaptive neuro-fuzzy controller based on emotional learning algorithm is proposed for the speed control of a brushless DC motor drive. In this paper, an experimental servo setup is used as a testing environment for an adaptive neuro-fuzzy control algorithm. The controller
structure is designed without a priori knowledge of the system parameters, and the controller scheme developed is tested experimentally with nonlinear load conditions.

II. MATHEMATICAL DESCRIPTION OF THE PERMANENTLY EXCITED DC MOTOR

The experimental setup [20] consists of two DC motors, which are connected by a mechanical clutch. The first motor is used for the control of the rotation speed or the shaft angle. The second one acts as a generator, by means of which nonlinear load conditions can be created (See Fig. 1).

![Fig. 1. Servo system setup](image)

The nomenclature of the symbols is given in Table 1. The transfer function of the overall system shown in Fig. 2 can easily be derived as follows:

\[
\Omega(s) = \frac{1}{C \Phi \left( 1 + T_M s + T_M T_A s^2 \right)} \frac{R_A}{1 + T_A s} M_L(s)
\]

where

\[
T_M = \frac{J R_A}{K_M C \Phi} \quad \text{and} \quad T_A = \frac{L_A}{R_A}
\]

![Fig. 2. Block diagram of the motor with load](image)

The numerical values used in this study are:
- Armature terminal voltage: 24V
- Rated torque: 0.096 Nm
- Moment of inertia of the system: 80.45x10^-6 kgm^2
- Armature inductance: 3mH
- Armature resistance: 3.13 ohm
- Back emf constant: 0.06 V/s
- Torque constant: 0.06 Nm/A

III. NEURO-FUZZY CONTROL SYSTEM

Many dynamic plants are characterized with uncertainties in terms of the structure and the parameters. These uncertainties cannot be adequately described by deterministic models. Therefore, conventional control approaches based on such models are unlikely to result in the required performance. Under such conditions the use of soft computing methodologies can be a valuable alternative. Not only do some plants in the industry have nonlinear characteristics, but also they are susceptible to internal and external disturbances. The time-varying nature of the plant may be interpreted as the uncertainties of the plant coefficients. This type of uncertainty can be dealt with using fuzzy sets. In this paper the FNS structure is used as the adaptive controller for the speed control of the DC motor.

The structure of the control scheme is shown in Fig. 3. Here \( y(k) \) is the output signal of the plant, \( g(k) \) is the set-point signal, \( e(k) \), \( \Delta e(k) \) and \( \sum e(k) \) are the error, the change in error and the sum of error, respectively. While \( D \) indicates differentiation operation, \( \sum \) indicates integration operation. The error, the change in error and the sum of error are the inputs to the FNS. Using these signals, the gradient based learning is used to tune the parameters of the FNS structure in a closed-loop fashion, and the IF-THEN rules of the controller are thus generated. The consequent parts of these rules result in the control signal to be applied to the plant.

In the neural networks or FNSs, the reference signal values and the desired output values of the network are used to train the system. However, the desired output values of the controller are not known in this study. Instead, using the error between the reference value of the control system and the current output value of the implemented system, \( \Delta = k_c (g(t) - y(t)) \), FNS is trained in a closed loop fashion. This is necessary since the desired output of the controller is not known! The supervised learning algorithm used is described in the next section.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( U_A )</td>
<td>Armature terminal voltage</td>
</tr>
<tr>
<td>( E )</td>
<td>Induced electromotive force</td>
</tr>
<tr>
<td>( I_A )</td>
<td>Armature current</td>
</tr>
<tr>
<td>( R_A )</td>
<td>Armature winding resistance</td>
</tr>
<tr>
<td>( L_A )</td>
<td>Armature winding inductance</td>
</tr>
<tr>
<td>( C \Phi )</td>
<td>Back emf constant</td>
</tr>
<tr>
<td>( K_M )</td>
<td>Torque constant</td>
</tr>
<tr>
<td>( \Omega )</td>
<td>Speed of the rotor</td>
</tr>
<tr>
<td>( M_L )</td>
<td>Load torque</td>
</tr>
<tr>
<td>( M_B )</td>
<td>Acceleration torque</td>
</tr>
<tr>
<td>( M )</td>
<td>The torque produced by the motor</td>
</tr>
<tr>
<td>( J )</td>
<td>Moment of inertia of the system</td>
</tr>
<tr>
<td>( T_A )</td>
<td>Electrical time constant</td>
</tr>
<tr>
<td>( T_M )</td>
<td>Mechanical time constant</td>
</tr>
</tbody>
</table>

![Table 1. NOMENCLATURE](image)
Fig. 3. Structure of Neuro-Fuzzy control system

Fig. 4. Structure of neuro-fuzzy inference system

IV. NEURO-FUZZY INFERENCE SYSTEM

The main element of the control system is the neuro-fuzzy controller, the structure of which is shown in Fig. 4.

Such structures combine the learning capabilities of neural networks with the linguistic rule interpretation of fuzzy inference systems. The synthesis of the neuro-fuzzy inference system for the controller includes the generation of the knowledge base that have rules in IF-THEN form. The problem consists of the optimal determination of the premise and the consequent parts through the training operation. In this study, TSK type fuzzy rules are used.

If \( x_1 \) is \( A_1 \) and \( \ldots \) and \( x_m \) is \( A_m \). Then \( y_j^l = \sum_{i=1}^{m} w_{ij}^l x_i + b_j^l \) (2)

where \( x_i (i = 1, \ldots, m) \) are the input variables, \( y_j (j = 1, \ldots, n) \) are the output variables which are linear functions. \( A_i^l \) is the fuzzy membership function for the \( l \)-th rule of the \( i \)-th input, \( w_{ij} \) and \( b_j \) are the parameters in the consequent part of rules.

The development of the FNS includes the determination of the proper values of the unknown coefficients in the antecedent and the consequent parts of each rule.

To come up with a control value based on the IF-THEN rules, the fuzzy system includes the fuzzification, the inference engine and the defuzzification functional blocks.

1) The fuzzification unit determines the membership degree of the crisp inputs.
2) The inference engine makes a decision on the base of these rules.
3) The defuzzification unit transforms the fuzzy results of the inference engine into a crisp output.

These functions are realized by the structure shown in Fig. 4 as follows: In the first layer, the system inputs are shown. In the second layer, each node corresponds to one linguistic term, the number of which is set by the expert of the problem domain. To describe the linguistic terms, Gaussian membership functions are used:

\[
\mu_{ij}(x_i) = e^{-(x_i-c_{ij})^2/\sigma_{ij}^2} \quad (3)
\]

where \( c_{ij} \) and \( \sigma_{ij} \) are the center and the width of the Gaussian membership function of the \( j \)-th term of \( i \)-th input variable, respectively. For each input signal entering the system, the membership degree is calculated using (3).

In the next layer (Layer 3), the number of nodes corresponds to the number of rules. Each node represents a fuzzy rule. To calculate the firing strengths of the rules, t-norm product operator is used.

\[
\mu_j(x_i) = \prod_j \mu_{ij}(x_i) \quad (4)
\]

The signals obtained by the use of (4) are the input signals for the consequent layer (Layer 4). The linear functions of the consequent part shown in (2) are calculated in Layer 5. Layers 6 and 7 are used for the defuzzification operation. In Layer 6, the sum of the product of the output signals of Layer 5 (the nominator part of (5)) and the sum of the output signals of Layer 3 (the denominator part of (5)) are determined. Then the “center of average” method is used for defuzzification operation. The output of the fuzzy system is thus determined as:

\[
u = \frac{\sum_{j=1}^{n} \mu_j(x_i) \cdot y_j}{\sum_{j=1}^{n} \mu_j(x_i)} ; \quad y_j = \sum_{i=1}^{m} w_{ij} x_i + b_j \quad (5)\]

Learning: The unknown parameters of the system are the parameters of the membership functions \( \varepsilon_{ij} \), \( \sigma_{ij} \) (\( i=1, \ldots, m \), \( j=1, \ldots, n \)) in the second layer, and the parameters of the linear functions \( w_{ij} \), \( b_j (i=1, \ldots, m, j=1, \ldots, n) \) in the fourth layer. To learn these parameters, a supervised learning algorithm is used.

At the first step, the output error is calculated:

\[
E = \frac{1}{2} \sum_{i=1}^{O} (u_i^d - u_i)^2 \quad (6)
\]

where \( O \) is the number of the output signals of the overall system (in the given case \( O=1 \)), \( u_i^d \) and \( u_i \) are the desired and the current output values of the overall system, respectively. The parameters \( w_{ij}, b_j (i=1, \ldots, m, j=1, \ldots, n) \) and \( c_{ij} \), and \( \sigma_{ij} \) are adjusted using the following formulas:

\[
w_{ij}(t+1) = w_{ij}(t) - \gamma \frac{\partial E}{\partial w_{ij}} ; b_j(t+1) = b_j(t) - \frac{\gamma}{\partial b_j} \quad (7)
\]
$c_{ij}(t+1) = c_{ij}(t) - \gamma \frac{\partial E}{\partial c_{ij}}; \sigma_{ij}(t+1) = \sigma_{ij}(t) - \gamma \frac{\partial E}{\partial \sigma_{ij}}$ (8)

where $\gamma$ is the learning rate. The derivatives in (7) are calculated as:

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial u} \frac{\partial u}{\partial y_j} \frac{\partial y_j}{\partial w_{ij}} = (u(t) - u^d(t)) \sum_{j=1}^{n} \mu_{ij} x_i, j = 1, .., n$$ (9)

$$\frac{\partial E}{\partial b_j} = \frac{\partial E}{\partial u} \frac{\partial u}{\partial y_j} \frac{\partial y_j}{\partial b_j} = (u(t) - u^d(t)) \sum_{j=1}^{n} \mu_{ij}, j = 1, .., n$$ (10)

The derivatives in (8) are determined by the following formulas:

$$\frac{\partial E}{\partial \sigma_{ij}} = \sum_j \frac{\partial E}{\partial u} \frac{\partial u}{\partial \mu_j} \frac{\partial \mu_j}{\partial \sigma_{ij}}$$

$$\frac{\partial E}{\partial c_{ij}} = \sum_j \frac{\partial E}{\partial u} \frac{\partial u}{\partial \mu_j} \frac{\partial \mu_j}{\partial c_{ij}}$$ (11)

where

$$\frac{\partial E}{\partial u} = u(t) - u^d(t), \quad \frac{\partial u}{\partial \mu_j} = \frac{y_j - u}{\sum_{k=1}^{m} \mu_{kj}}$$ and

$$\frac{\partial \mu_j}{\partial \mu_{ij}} = \prod_{k=1}^{m} \mu_{kj}$$ (12)

where $i=1,..,m, k=1,..,m, j=1,..,n$.

The adjustment of the membership functions of the input layer is carried out by learning of the unknown coefficients $c_{ij}$ and $\sigma_{ij}$. The following formulas can be used for this purpose:

$$\frac{\partial \mu_{ij}(x_i)}{\partial c_{ij}} = \mu_{ij}(x_i) \frac{2(x_i - c_{ij})}{\sigma_{ij}^2}$$

$$\frac{\partial \mu_{ij}(x_i)}{\partial \sigma_{ij}} = \mu_{ij}(x_i) \frac{2(x_i - c_{ij})^2}{\sigma_{ij}^3}$$ (13)

One important problem in learning algorithms is the convergence. It is well known that the convergence of the gradient descent method depends on the selection of the initial value of the learning rate. This value is usually selected in the interval $[0,1]$. While a large value of the learning rate may lead to unstable learning, a small value of the learning rate results in a slow learning speed. In this paper, an adaptive approach is used for updating the learning rate which is started with a small value. During the learning operation, $\gamma$ is increased if the value of the change of error $\Delta E=E(t)-E(t-1)$ is positive, and decreased if it is negative. This strategy ensures a stable learning for the FNS, guarantees the convergence and speeds up the learning.

In summary, the update process of the fuzzy neural network parameters is carried out using (6)-(13).

V. EXPERIMENTAL RESULTS

In order to evaluate the performance of the proposed control algorithm (FNS), experimental studies are carried out on the set-up shown in Fig. 1 and compared with those obtained with a Proportional-Integral (PI) controller. The sampling period is set to 20 ms for all the experiments. The speed of the motor and the load torque are scaled to the range of [-1,1]. In order to test the efficiency and the accuracy of the proposed controllers, three different types of load are used. Fig. 5 represents the speed responses of the motor for FNS and PI controllers. The corresponding load condition is shown in Fig. 6, which indicates that the load torque starts with a value of 0.2, and increases suddenly to 0.5 on 7th s, and then comes back to 0.2 on 14th s.

In order to initialize $\mu_{ij}$ and $b_j$ matrices in (5) and $c_{ij}$ and $\sigma_{ij}$ in (3), first the simulation model of the setup is run for 100 epochs. The final values obtained from the training operation are used as the initial values in the real time experiments. The initial value for the learning rate for both simulations experiments is set to 0.0001. It is to be stated that the coefficients of the PI controller are tuned such that the best available performance (minimum overshoot and settling time) can be obtained for every different load conditions, i.e. $K_P = 1$ and $K_I = 12$, $K_P = 1$ and $K_I = 5$, and $K_P = 1$ and $K_I = 10$ in Fig. 5, Fig. 9, and Fig. 11, respectively.

As can be seen from Fig. 5, FNS adapts its parameters when the load of the motor changes suddenly on 7th s and on 14th s. Fig. 7 and 8 show the control signal and the motor armature current, respectively.
In Fig. 9 the speed response of the motor under the sinusoidal load condition (shown in Fig. 10) is given and Fig. 11 shows the speed response of the motor under a load condition which is proportional to the square of the speed, i.e. \( M_L = 1.5(Speed)^2 \), requiring a much lower torque at low speeds than at high speeds. In both cases, FNS controller gives better performance than the PI controller.

In order to make a quantitative comparison between FNS and PI controllers, the following function is defined:

\[
SER = \sum_{i=1}^{N} e^2(i) \quad (14)
\]

where \( SER \) is the square of the error values at each time step of the control algorithm, and \( N \) is the number of the samples.

In Table II, SER values for 3 different experiments are given. The load type 1, the load type 2 and the load type
3 refer to the step change in load, the sinusoidally changing load, and the load proportional to the square of the speed, respectively.

VI. CONCLUSION

In this paper, the design of an adaptive neuro-fuzzy control architecture is presented. The structure of the neuro-fuzzy system is described and the appropriate learning algorithms are derived. The learning capability of the neuro-fuzzy inference system allows it to deal with the non-stationary plants, which cannot be efficiently controlled by the conventional controllers. This statement is confirmed through simulations and experiments by using the proposed structure. In order to have a basis for comparison, the same experiments are carried out with a well-tuned PI controller. A comparative evaluation of the results demonstrates the efficacy of the presented approach.

ACKNOWLEDGMENT

The authors would like to acknowledge the financial support of the Bogazici University Research Fund with the project number 08A204, the TUBITAK through Project No. 107E284 and the Program 2221.

REFERENCES