PID Neural Network Based Speed Control of Asynchronous Motor Using Programmable Logic Controller

Vahdet Ayhan MARABA[1], Ahmet Emin KUZUCUOGLU[2]
[1]*Haydarpasa Vocational High School, Department of Information Technology, Istanbul, Turkey*
[2]*Marmara University, Faculty of Technical Education, Department of Electronics and Computer Education, Istanbul, Turkey*
kuzucuoglu@marmara.edu.tr

***Abstract*—This paper deals with the structure and characteristics of PID Neural Network controller for single input and single output systems. PID Neural Network is a new kind of controller that includes the advantages of artificial neural networks and classic PID controller. Functioning of this controller is based on the update of controller parameters according to the value extracted from system output pursuant to the rules of back propagation algorithm used in artificial neural networks. Parameters obtained from the application of PID Neural Network training algorithm on the speed model of the asynchronous motor exhibiting second order linear behavior were used in the real time speed control of the motor. Programmable logic controller (PLC) was used as real time controller. The real time control results show that reference speed successfully maintained under various load conditions.**

***Index Terms*—control, neural network, PID, PIDNN, PLC.**

I. INTRODUCTION

The basic purpose of control systems is to keep system behavior on the desired value. The controller located in a closed loop control system generates the control signal required in order to keep the system output on the desired value.

Selection of the controller to be used in the control of systems and detection of parameters are fundamental tasks in classical control methods. Parameters of the controller to be devised for the system that will be checked are found through a model that represents the behavior of that system. Controller parameters that are found cannot always provide the stability of a system to the desired extent due to factors such as modeling mistakes of system, changes in the parameters of the controlled system and disruptive effects. In spite of all of these problems in classical control methods, Artificial Neural Networks started to be used in the field of control since they have the ability to learn and generalize and there is no obligation to form a mathematical equation [1].

Most of the systems used in industry today are non linear time-delay behavior. These systems have excessive overshoot and high settling times and are not stable. It is highly difficult and demanding to design a controller by using through classic methods in such systems. Controllers can be designed by using various methods if mathematical models or transfer functions of systems can represent the system behavior very well. However, it is rather difficult to have mathematical models of such systems in practice.

Classical PID controllers are immensely preferred in many areas of industrial control, especially in chemical industry due to their simple structure and high durability. Although PID controllers are used for controlling many systems; it is difficult to find optimum parameters (K_P, K_I, K_D) in the control of time-delay and nonlinear systems [2], [3]. Different intelligent approaches have successfully been integrated to PID controller. The fuzzy supervisor was used to improve the results of the PI regulator to control of induction machine speed [4]. A wavelet adaptive PID controller, based on reinforcement learning is used to control the wind energy conversion system [5].

PID-Neural Network controllers perform adaptive control. Uncertainties in systemic and environmental factors ensure that the controller performs a better behavior by means of adaptive control. However, there are problems to be solved in practice. These can be evaluated as the disadvantages of controller. The main problem is the slow learning rate and slow approximation to the reference value and the uncertainties in the parameters of the controller [2].

A PIDNN controller includes the advantages of PID algorithm and neural network. This controller is not a hybrid structure consisting of artificial neural networks and PID controller [6]. PID algorithm exists in neurons located in the neural structure as an activation function.

PID-Neural Network is a new kind of controller. There are P-proportional, I-Integral, D-derivative neurons in the network structure and the connection weights arising among these neurons are updated by using back-propagation training algorithm according to the error value propagated through the system.

PLC is widely used in industrial process control. This microprocessor based controllers work very well at factories in various environment conditions.

II. STRUCTURE OF PIDNN CONTROLLER

A. General Structure of PIDNN

PID-Neural Network controllers are a new type controller in which artificial neural networks are used with PID algorithm. As it is known, in classical PID controllers, a control signed is found by multiplying the error value that occurs in the system with coefficients like K_P, K_I, K_D. However, in the controller that we analyze, PID structure is

Digital Object Identifier 10.4316/AECE.2011.04004

found in the neurons of the network.

The controller consists of three layers. In input layer, there are neurons that give the reference value and system output values as input into the system. In the middle or hidden layer, there are the neurons which perform PID functions. P neuron performs the proportional function, I neuron performs the integral function, and D neuron performs derivative function. In the output layer, there is the neuron in which the control signal that the controller will give to the system is calculate.

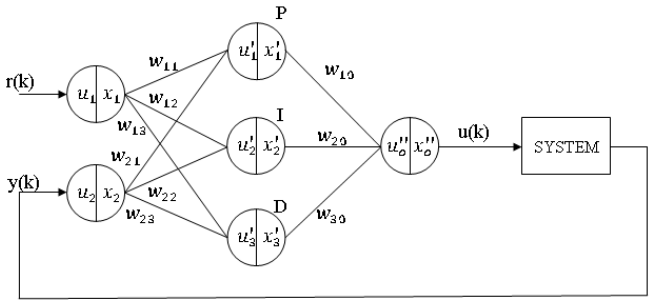

Figure 1. Structure of PIDNN controller [2]

The controller consists of two parts. First one is the forward direction calculation in which the control signal is calculated; the second one is the training stage where connection weights are adjusted by using the error value propagated through the system and back propagation algorithm [7]. The PIDNN structure for single-input-single-output systems is shown in Fig. 1.

B. Feed Forward Calculation

Input Layer

There are two neurons in this layer. Input-output functions of these neurons are demonstrated in Eq.(1).

$$x_i(k)=\begin{cases}1 & u_i(k)>1\\ u_i(k) & -1\le u_i(k)\le 1\\ -1 & u_i(k)<-1\end{cases} \quad (1)$$

i=1,2 and k = number of samples.

Hidden Layer

This layer contains the neurons that perform the basic function of controller. The values calculated by multiplying the output values of neurons in the input layer and the connection weights between the input and hidden layers come to the neurons in this layer and are processed in the neurons conducting the PID algorithm. Inputs formed in the neurons of the hidden layer are calculated like in Eq.(2).

$$u'_j(k)=\sum_{i=1}^{2} w_{ij}.x_i(k) \quad (2)$$

j=1,2,3 w_{ij} is the connection weights formed between the input and the output layer.

After the values formed in the inputs of hidden layer neurons are calculated, the functions of neurons in the forward direction calculation stage are demonstrated in Eq.(3-5).

P-Neuron;

$$x'_1(k)=\begin{cases}1 & u'_1(k)>1\\ u'_1(k) & -1\le u'_1(k)\le 1\\ -1 & u'_1(k)<-1\end{cases} \quad (3)$$

Expression of the output value of P Neuron located in the hidden layer is demonstrated in Eq.(3).

I-Neuron;

$$x'_2(k)=\begin{cases}1 & x'_2(k)>1\\ x'_2(k-1)+u'_2(k) & -1\le x'_2(k)\le 1\\ -1 & x'_2(k)<-1\end{cases} \quad (4)$$

Expression of the output value of I neuron located in the hidden layer is demonstrated in Eq.(4). Accordingly, the neuron output is found by adding the value coming to the input of the neuron and the previous value of output.

D-Neuron;

$$x'_3(k)=\begin{cases}1 & x'_3(k)>1\\ u'_3(k)-u'_3(k-1) & -1\le x'_3(k)\le 1\\ -1 & x'_3(k)<-1\end{cases} \quad (5)$$

Expression of the output value of D neuron located in the hidden layer is demonstrated in Eq.(5). Accordingly, the neuron output is found by adding the value coming to the input of the neuron and the previous value of output.

Output Layer

Values obtained at the outputs of hidden layer neurons come to the input of the neuron in the output layer by being multiplied with connection weights between the hidden and output layers.

$$u''_0(k)=\sum_{j=1}^{3} w_{j0}.x'_j(k) \quad (6)$$

The net total calculated with Eq.(6) passes through the output layer and forms the control signal. Input-output function of output layer is demonstrated in Eq.(7). Accordingly, the output neuron is equally propagated to the output, on condition that the value at its input is between 1 and -1 values.

$$x''_0(k)=\begin{cases}1 & u''_0(k)>1\\ u''_0(k) & -1\le u''_0(k)\le 1\\ -1 & u''_0(k)<-1\end{cases} \quad (7)$$

When we look at the network structure of PIDNN controller, we can see that the multi-layer artificial neural network acts as a classical PID controller by using appropriate connection weights. If we choose the connection weights between the input layer and the hidden layer in the PIDNN structure as in Eq.(8), connection weights between

$$W_{1j}=+1, \quad W_{2j}=-1, \quad j=1,2,3 \quad (8)$$

the output layer and the hidden layer are the coefficients found in classical PID structure. These network parameters are demonstrated in Eq.(9).

$$W_{10}=K_P, W_{20}=K_I, W_{30}=K_D \quad (9)$$

According to these values, values coming to the inputs of neurons in the hidden layer are like the values given in Eq.(10).

$$\begin{aligned} U'_1 &= W_{11}X_1+W_{21}X2=r-y=e,\\ U'_2 &= W_{12}X_1+W_{22}X_2=r-y=e,\\ U'_3 &= W_{13}X_1+W_{23}X_2=r-y=e, \end{aligned} \quad (10)$$

According to the formulation in Eq.(10), the error value that is the difference between system reference and system output is applied to the inputs of the neurons located in the hidden layer due to the values of connection weights. That is to say, error values come to the inputs of the neurons located in the hidden layer. The value found after the neurons passed through activation functions, which are P, I, D neurons, are given in Eq.(11).

$$X_1^{'} = U_1^{'} = e,$$

$$X_2^{'} = \int_0^t U_2^{'} dt = \int_0^t e dt, \tag{11}$$

$$X_3^{'} = \frac{dU_3^{'}}{dt} = \frac{de}{dt},$$

Accordingly, network output is found by multiplying error values with the connection weights between the hidden layer and the output layer. This expression is given Eq.(12).

$$X_0^{''} = U_0^{''} = \sum_{j=1}^{3} W_{j0} X_j^{'} = W_{10} X_1^{'} + W_{20} X_2^{'} + W_{30} X_3^{'} \tag{12}$$

In conclusion, choosing the connection weights between the input layer and the hidden layer according to the values demonstrated in Eq.(8) causes the PIDNN network structure to become a classical PID. When the activation functions of neurons located in latent layer are considered according to Eq.(11), controller network output becomes like shown in Eq.(13).

$$X_0^{''} = K_P e + K_I \int_0^t e dt + K_D \frac{de}{dt} \tag{13}$$

As a result, the designed network structure exhibits the behavior of a classical PID depending on the values of the connection weights [2].

This feature of PIDNN controller makes it easy to use in practice. Artificial neural networks are not used for system control in practice due to the uncertainties in the initial weights and the failure to ensure stable operation of the system. We can find appropriate initial weights by operating the PIDNN like a classical PID control at the first time of operation. After these weights are chosen, system response and control performance can be improved by operating the backpropagation training algorithm [8].

III. General Structure of The Controlled System

The asynchronous motor system to be controlled is composed of 3 main parts as it is seen Fig. 2a. Mechanical part of the system is designed so as to act as fixed brake by charging the AC motor operating at fixed speed with fixed load. In this part, there is a 200W DC generator, 4 spot lamps with 12V 50W power, and the relay card. On the left hand side there is a 3-phase 250W AC motor and Micromaster-420 driver and a tacho generator that generates a voltage according to speed of AC motor. Siemens S7-300 CPU313C-2 DP programmable logic controller and SM334 Analog I/O module are used as the control unit [9]. As it is seen in Fig. 2b, the controller algorithm that works on S7-300 PLC environment communicates with the system through analog and digital input-output modules. The spot lamps connected to the DC generator which physically loads the asynchronous motor are loaded or unloaded analogously. The tacho generator that conveys instantaneous information on revolution frequency in block diagram is directly connected to the PLC and the control signal generated by software is sent to the driver through the analog I/O module of PLC.

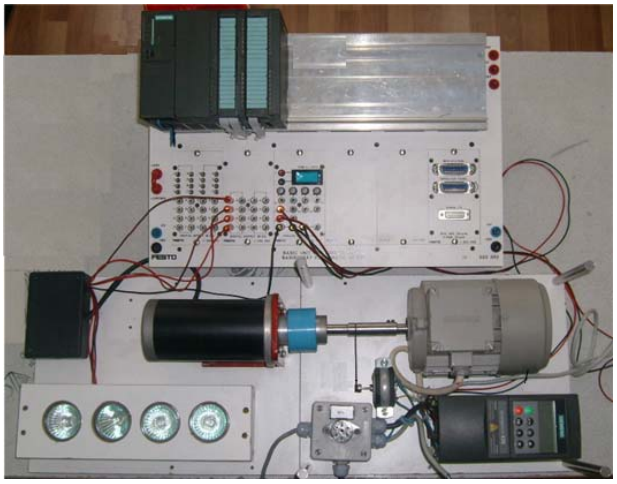

Figure 2a. Experimental setup of asynchronous motor system

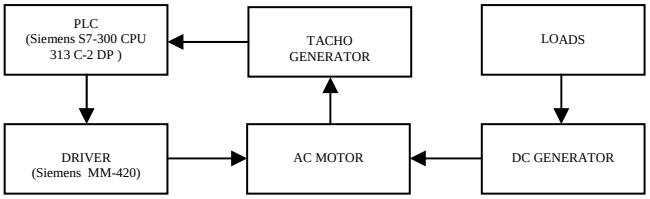

Figure 2b. Block diagram of the system

IV. Speed Control Studies of AC Motor

A. System Model

Generally, system modeling means measuring the inputs, outputs and finding the physical system variables with model approach. The input-output data set forming a system model is a data group that contains values sampled in sufficient numbers [10]. To collect the input-output data, National Instruments PCI-6024E data acquisition card and PC are used. The input-output data set forming was completed in two stages. In the first one, the control signal was varied between 0-10V in step function form. In the second stage, the control signal was varied randomly between 0-10V and the speed of the AC motor was recorded in both stages for 500 seconds. The sampling time was used as 0.1sec. The input-output data set was formed by using these data. This data set was used in Matlab System Identification Toolbox to get the speed model of AC motor. ARX (Auto Regressive eXpgenous) model was chosen and the model is given in Eq.(14).

$$A(q)y(t) = B(q)u(t) + e(t) \tag{14}$$

y(t) is the output and u(t) is the input. A(q) corresponds to poles that are common for the dynamic model and noise model. B(q) represents the contributions of inputs to predicting all output values [11]. ARX model order was chosen as two. After applying the data set to program,

$$A(q) = 1 - 1.283q^{-1} + 0.369q^{-2}$$

$$B(q) = 0.01103q^{-1} + 0.03599q^{-2}$$

A and B polynomials have been determined. The difference equation of the model can be shown as in Eq.(15).

$$y(k+1) = 0.01103*u(k) + 0.03599*u(k-1) + 1.283*y(k) - 0.3969*y(k-1) \quad (15)$$

Or speed model of the AC motor can be shown as transfer function in discrete time in Eq.(16).

$$\frac{y(z)}{u(z)} = \frac{0.01103z + 0.03599}{z^2 - 1.283z + 0.3969} \quad (16)$$

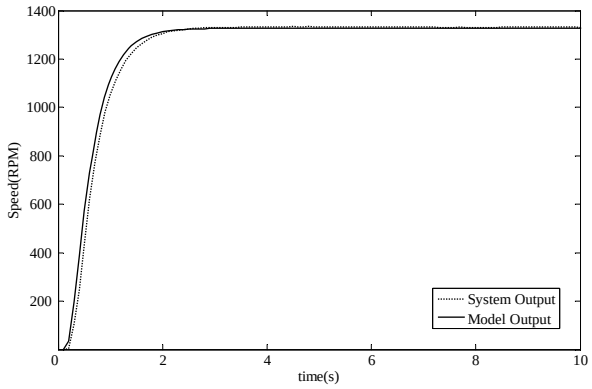

Figure 3. Open loop comparision of system and model output for 1300RPM

In Fig.3 the ARX model output and the speed of AC motor which is taken from taco generator was compared. Clearly the model is good. It catches relevant features of motor in the rising time and the settling time. In Fig.4 the real time system output and the model output is compared in various speeds. Some minor differences can be accepted in transitions of steps. In Fig.5 the error between the model output and the real system output is given.

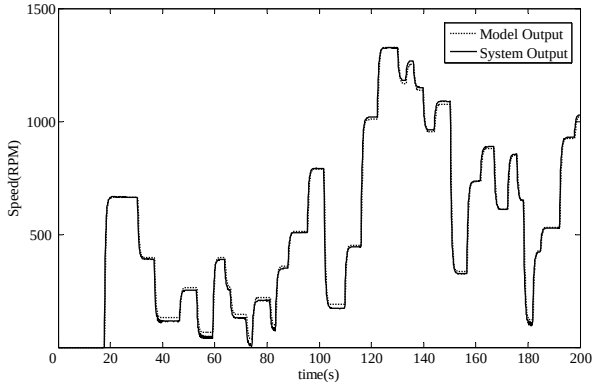

Figure 4. Various speed results of system and model output

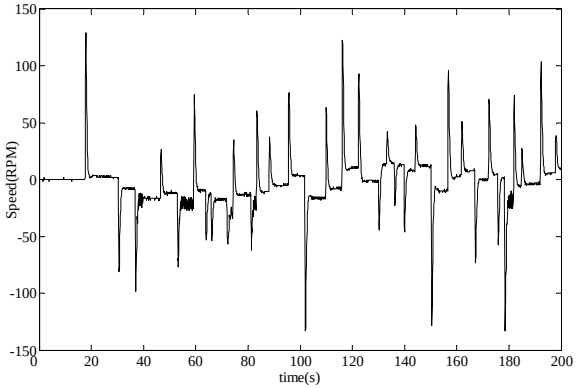

Figure 5. Errors between system and model output

After the difference equation of asynchronous motor in discrete time is obtained, the process of improving the system response through this model is performed at the training stage of the controller. The importance of artificial neural networks at this point is the use of back propagation algorithm aimed at decreasing the mean square of errors. Structure of the controller is not complex and classical PID functions are carried out in the network. There are parameters to be determined before starting the training process; which are learning rate, number of trainings and the initial value of weights. Achievement of optimization is closely related to choosing the initial values of such parameters as values appropriate to the system.

The most important part of PIDNN controller algorithm is the minimization of the cost function. As it is seen in Fig.6, cost function specified as the average of square errors decreased after each of training iteration. Error value was high in the first training step, however, the parameters applied after each update caused the error in the system response to decrease compared to the previous error. Whether the training is quick or slow does not mean that the appropriate parameters will be found more accurately.

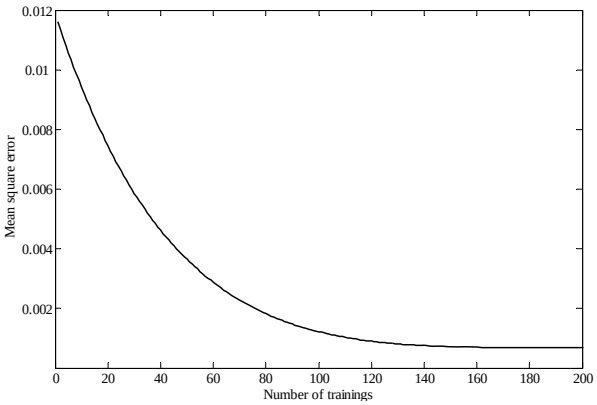

Figure 6. Cost Function

Learning rate is a factor that determines the training rate. As it was told before, there is no predefined rule for the initial value of the learning rate; it can change depending on the implementation. If the training is continued after the 200^{th} iteration, the system value gets unstable and oscillates. Therefore maximum number of trainings can change depending on the implementation. Termination of the training is decided on when the criterion determined as the cost function fall below a value determined by the user, or the user can examine the system response at the end of each iteration and then decides to stop the training.

Connection weights obtained at the end of the training were used in the real time operation of the system. Rise time of the system response obtained as a result of training is short, which means the system output has rapidly approached the related reference value. Overshoot amount is low, and steady state error is close to zero. Trained system response and open loop system response are compared in Fig. 7. The settling time was 2 sec. in open loop system, which decreased to around 1 sec. using the controller. Fig. 8 shows the values of the controller signal obtained from the output of the controller network.

In the real time study, the neuron at the PIDNN output or the controller network output is limited between 0 and 1 or -1 and 1 as it is told in the output layer section. Output layer, however, no restriction criteria is put at the network output since the controller is operated on the model.

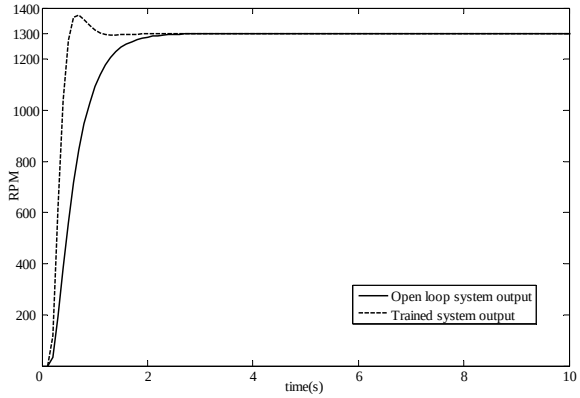

Figure 7. System responses according to open loop and optimized parameters

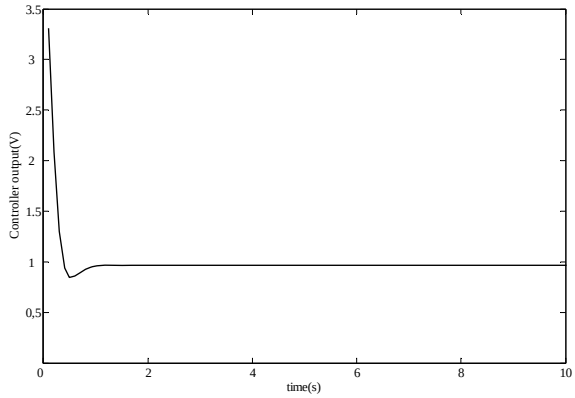

Figure 8. Optimized controller network output

Controller network output is seen in Fig. 8 without any restrictions. The controller network output in the real time study should be limited between 0-10 Volts; otherwise the related system could be given a control signal above 10 Volts.

B. System Responses to Step Input Under Programmable Logic Controller

Connection weights obtained at this stage of the study were used in the real time operation of the system. These weights were used in PLC. PLC was programmed using SCL [12]. Five reference inputs were applied to the PLC controlled system. The system outputs and the control signals for these reference inputs are given in Fig.9a and Fig.9b.

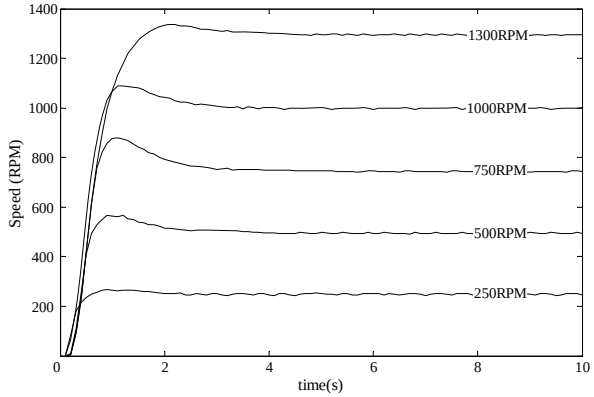

Figure 9a. The response of PLC for five different step input

As it can be seen at control signal output, when the system response approaches the related value in reference speed graphic, the control signal ceased the maximum drive and continued to generate the signal required for the related reference.

One of the PC and PLC controlled real system responses was chosen. PC controlled system output and PLC controlled system output were compared for reference values of 750 RPM in Fig.10. The response of PLC controlled system is slightly slow and the steady state error is slightly high according to PC controlled system.

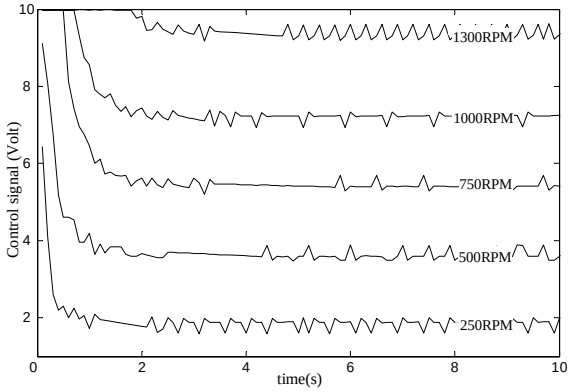

Figure 9b. The control signals of PLC for five different step input

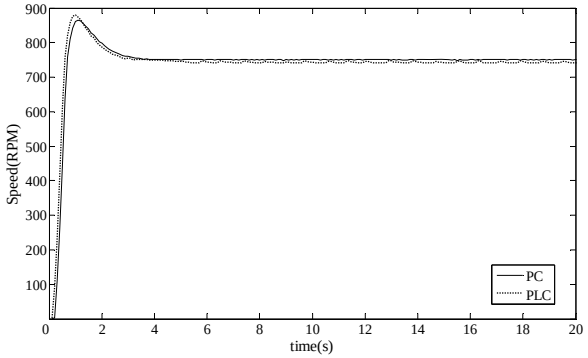

Figure 10. Comparison of PC and PLC controlled systems output for 750RPM

C. System Response Under Load

A DC generator was connected in order to apply load to the shaft of the asynchronous motor. DC generator runs 4 lamps 50W each acting like generator. Each lamp is switched on one by one and, the control signals and the speed of the AC motor were observed. Two load conditions result are introduced in Fig.11. A rapid decrease of 30RPM was observed on shaft of AC motor operating at 1000RPM when two spot lamps of 50W were turned on. This decrease of speed recovered quickly by simultaneously increasing the control signal and the speed of the motor was held on the reference speed. When the motor was unloaded, an increase of the speed at same rate was observed, but the controller turned back to its reference speed value (Fig.12).

In Fig.11, four spot lamps of 50W each were simultaneously turned on, and a load of 200W was generated. When the motor is loaded, number of revolutions decreased by approximately 35 RPM and the controller offset the load by increasing the control signal. When the motor was unloaded, number of revolutions increased at the same rate and the controller decreased the control signal in order to get the related reference speed at system output (Fig.12).

V. RESULTS AND DISCUSSION

Optimization process was completed by decreasing the cost function below the error tolerance value via the parameters that were found. With these coefficients, the rise

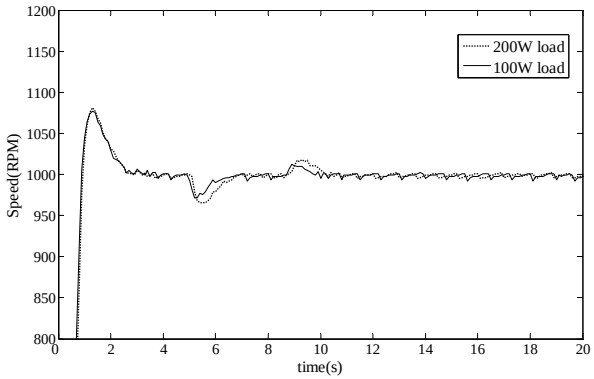

Figure 11. The response of the PLC controlled system to loads for 100W and 200W

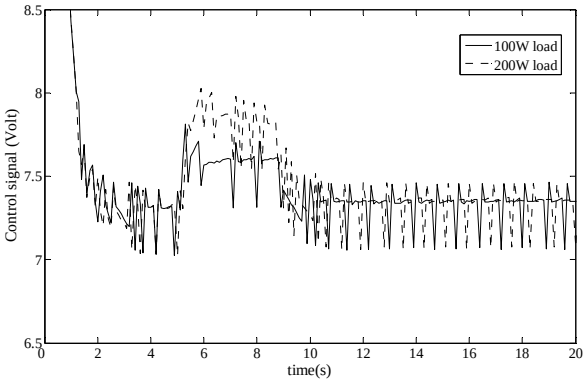

Figure 12. Control signals for loaded of the system of the PLC controlled

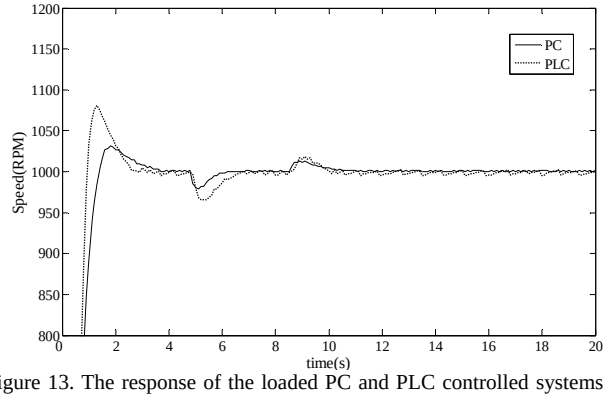

Figure 13. The response of the loaded PC and PLC controlled systems for 200W

TABLE I
PARAMETERS OF SPEED RESPONSE FOR PC CONTROLLER

	PC			
	Tr(s)	Ts(s)	Os(%)	SSE(%)
250RPM	0.35	2.10	6.03	<% 2.0
500RPM	0.34	2.50	10.50	<% 2.0
750RPM	0.37	2.90	15.30	<% 0.5
1000RPM	0.49	2.70	8.70	<% 0.5

TABLE II
PARAMETERS OF SPEED RESPONSE FOR PLC CONTROLLER

	PLC			
	Tr(s)	Ts(s)	Os(%)	SSE(%)
250RPM	0.32	2.10	6.40	<% 2.0
500RPM	0.31	2.50	14.00	<% 2.0
750RPM	0.44	4.00	17.00	<% 1.0
1000RPM	0.50	3.20	8.90	<% 0.5

time of the system in closed loop was decreased from 2 sec. to around 1 sec. Data obtained from one of the real time studies at the reference speed of 1000RPM, and the data obtained from the simulation studies conducted on the model yielded approximately similar results according to control criteria such as rise time, settling time and overshoot amounts. Response of the AC motor to staircase driving was quick and efficient. When load was applied to the system, controller increases its drive effect and responded to the load.

The response of the PC and The PLC controlled system is shown in Fig.13. The main difference is in overshoot amount. The difference is approximately 50RPM. But response to loading and unloading is almost same. The overshoot and the loading response of PLC is mainly related to sampling time. The ripple on the response is related to resolution of the PLC which is 8 bits. The PC controller's DAQ cards' resolution is 12 bits. The PLC controller doesn't give any response until 39mV changes in the system. But the PC controller gives response after 2.44mV changes of the system. If the overall system's conditions is considered, the PLC's response to sudden changes is slightly slow than PC.

The performance comparison of PC and PLC is given in Table I and II. The comparison was made in four main parameters. The rise time (Tr(s)), the setting time (Ts(s)), the overshoot (Os(%)) and the steady state error (SSE(%)). The values of parameters were shown that the PLC can also be used in various working conditions very efficiently.

VI. CONCLUSION

In this study, initial values of controller parameters such as the number of trainings, learning rate and initial weights in the PIDNN structure used were given and the controller parameters to be used in real time application was quickly found as a result of the operation. In this way, controller parameters of systems with very different structures can be found via PIDNN training algorithm, and real time control processes can be conducted efficiently using PC and PLC.

REFERENCES

[1] Dandil B., Plant control by aid of artificial neural networks, MSc Thesis, Firat University Institute of Applied Sciences, 1997, (in Turkish).

[2] Shu, H., Pi, Y., "PID neural networks for time-delay systems", Computers and Chemical Engineering, Vol. 24, Issues 2-7, pp 859-862, 2000,.

[3] Bekiroglu, N., Ozcira, S., "Observerless Scheme for Sensorless Speed Control of PMSM Using Direct Torque Control Method with LP Filter", Advances in Electrical and Computer Engineering, Vol. 10, Number 3, pp. 78-83, 2010.

[4] Laroussi, K., Zelmat, M., Rouff, M. "Implementation of a Fuzzy Logic System to Tune a PI Controller Applied to an Induction Motor", Advances in Electrical and Computer Engineering, Vol. 9, Number 3, pp. 107-113, 2009.

[5] Sedighizadeh, M., Rezazadeh, A., "A modified Adaptive Wavelet PID Control Based on Reinforcement Learning for Wind Energy Conversion System Control", Advances in Electrical and Computer Engineering, Vol. 10, Number 2, pp. 153-159, 2010.

[6] Shu, H., Guo, X., Shu, H., "PID neural networks in multivariable control systems", International Symposium on Intelligent Control, Vancouver, Canada, 2002.

[7] Shu, H., Guo, X., "Decoupling Control of Multivariable Time-Varying Systems Based on PID Neural Network", 5th Asian Control Conference, Melbourne, Australia, 2004.

[8] Shu, H., Pi, Y., "Decoupled Temperature Control System Based on PID Neural Network", ACSE Conference, Cairo Egypt, 2005.

[9] Siemens S7-300 Module Data, A5E00105505-06, 08/2009.

[10] Ronco E, Gawthrop P J, Neural networks for modeling and control, Centre for System and Control Department of Mechanical Engineering University of Glasgow, Technical Report csc97008, 1997.

[11] Lennart Ljung, System Identification: Theory for the user, 2nd edition (Prentice Hall PTR, 1999).

[12] Siemens S7-SCL V5.3 for S7-300/400 Manual, Edition 01/2005.