Proceedings of the 7th Mediterranean Conference on Control and Automation (MED99) Haifa, Israel - June 28-30, 1999

Development of a Self-tuning PID Controller based on Neural Network for Nonlinear Systems

Woo-yong Han[1], *Jin-wook Han*[2], *Chang-goo Lee*[3]

Dept.of Electrical Engineering, Jeonju Technical College, Korea
Dept of Electronics Engineering, Chonbuk Nationl University, Korea
Faculty of Electronic and Information Engineering, Chonbuk National University, Korea

ABSTRACT

The key point of this research is to develop a fast tracker for time-varying nonlinear systems which previous knowledge (i. e., dynamic equations) about the plant were not known. In order to carry out this research goal, this paper suggests a novel error self-recurrent neural networks, and develops a fast on-line learning algorithm by using the recursive least squares method. The new neural networks are considerably faster than the backpropagation algorithm and have advantages of being less affected by poor initial weights and learning rate. Nonlinear adaptive PID controller based on these neural networks has been derived and tested for the fast tracking problem in a robot manipulator.

I. Introduction

Neural networks have been one of the most interesting topics in the control community because they have the ability to treat many problems that cannot be handled by traditional analytic approaches. In general, feedforward multilayer neural networks are the most prevalent neural network architecture for identification and control applications [1][2]. A widely used training method for a feedforward multilayer neural networks (MNN) is the backpropagation (BP) algorithm developed by Rumelhart et al. [3]. The standard BP learning algorithm has several limitations. Most of all, a long and unpredictable training process is the most troublesome, for example the rate of convergence is seriously affected by the initial weights and the learning rate of the parameters. Many researchers have proposed the modifications of the classical BP algorithm [4][5]. Recently another modified algorithm was derived by Scalero and Tepedelenlioglu as an alternative to the BP algorithm. It uses a modified form of the BP algorithm to minimize the mean square error between the desired output and the actual output with respect to the summation output [6]. Even though the performance of this new algorithm overwhelms the BP method, it is not a stable learning algorithm in practical real-life applications. Thus, the faster and more stable learning neural network is required, which is indeed the main purpose of this paper. For solving those problems addressed above, the novel error self-recurrent neural networks are presented, and the desired outputs of the hidden layers are obtained by the nearly optimal learning algorithm. A new neural network is considerably faster than the BP algorithm and has advantages of being less affected by poor initial weights and learning rate. Nonlinear adaptive PID controller based on these neural networks has been derived and tested for the fast tracking problem in a robot manipulator model.

[1] Email : wyhan@www.jtc.ac.kr
[2] Email : modesty@icrn.chonbuk.ac.kr
[3] Email : changgoo@moak.chonbuk.ac.kr

II. Error Recurrent Neural Networks

It is shown in Fig. 1 and 2 that all node input x_{ik} and summation outputs y_{ik} were specified. The problem would be reduced to a linear problem, i. e., a system of linear equations that relates the node inputs x_{ik} to summation output y_{ik} through weight vector w_{ik} .

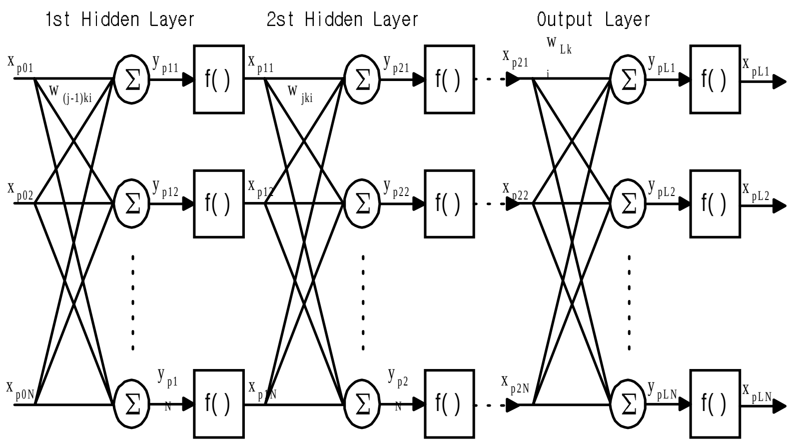

Fig. 1 Structure of a Multilayer Feedforward Neural Network

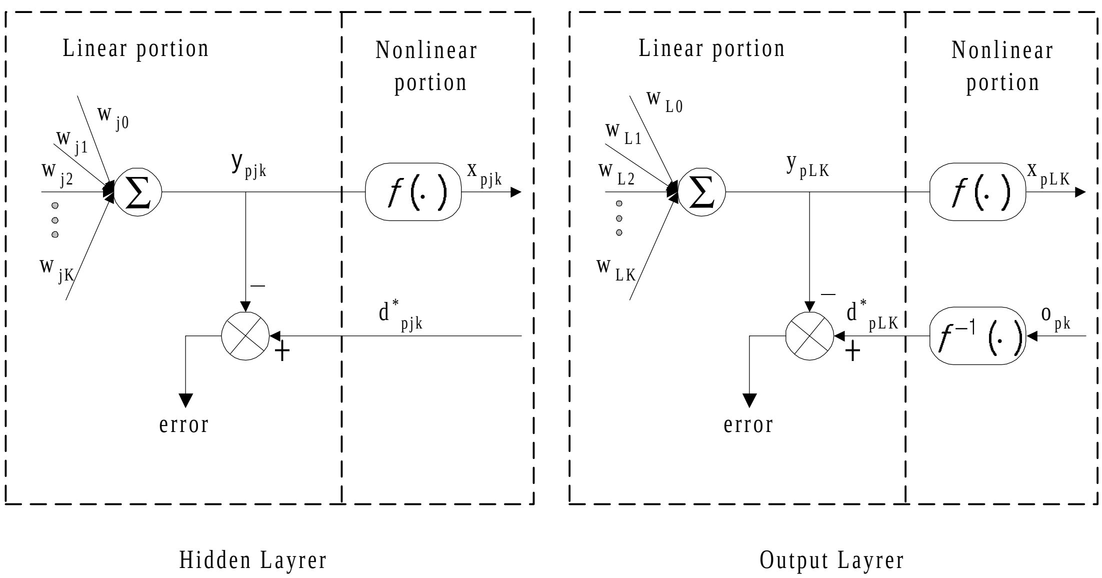

Fig. 2 Linear and nonlinear portion of the neurons in the hidden and output layers

The learning technique with Kalman filtering of MNN builds upon the partitioning into linear and nonlinear parts proposed by Scalero et al [6]. First, the desired summation outputs d_{ik} of the nonlinear portion were calculated, which update and optimize the weights of the linear part between the summation outputs y_{ik} and the inputs x_{ik} by using Kalman filtering method. The Scalero's algorithm can be summarized by the following steps:

1) Calculate the desired summation outputs and δ_{ik} is obtained by using the steepest descent method.

$$d_{Lk} = f^{-1}(o_k) \qquad (1)$$
$$d_{jk} = y_{jk} + \mu\delta_{jk} \qquad (2)$$

where μ is the learning rate, $f(\bullet)$ is activation function, and j, k, and L identify the layer, node, and output layer, respectively. o_k is the desired value of output layer.

2) For each layer j, from 1 through L, calculate the Kalman gain and update the covariance matrix.

$$\mathbf{k}_j = P_j x_{j-1} / (\lambda + x_{j-1}^T P_j x_{j-1}) \qquad (3)$$
$$P_j = (I - k_j x_{j-1}^T) P_j / \lambda \qquad (4)$$

where x_{j-1} is the input vector of j-th layer, and λ is forgetting factor.

3) Update the weights.

$$\mathbf{w}_{Lk} = \mathbf{w}_{Lk} + \mathbf{k}_L (d_{Lk} - y_{Lk}) \qquad (5)$$
$$\mathbf{w}_{jk} = \mathbf{w}_{jk} + \mu \mathbf{k}_j \delta_{jk} \qquad (6)$$

2.1 Error Self-Recurrent Neuron Model

In conventional neuron model shown Fig. 3(a), the bias or threshold given externally has the effect of increasing or decreasing the net input of the activation function respectively [7]. However, it is difficult to determine their magnitudes and signs, also it is hard to explain physical phenomena.

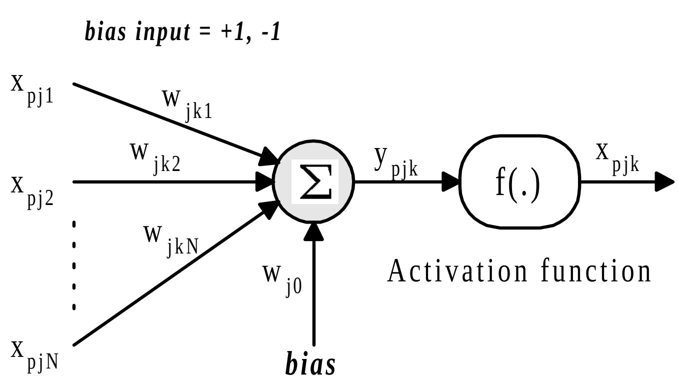

Fig. 3(a) Conventional Neuron Model

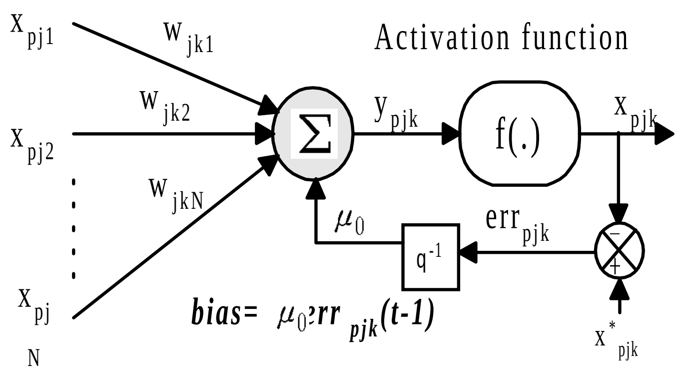

Fig. 3(b) Error Recurrent Neuron Model

In a novel neuron model, 1) input for bias or threshold takes the values of one step ahead error signal($x_{ik}(t-1)$) between the desired output and the model output, and 2) synaptic weight with fixed value is added to the recurrent error signals. This architecture is depicted in Fig. 3(b). The new neuron model has an automatic decision on sign and magnitude for threshold or bias value instead of manually selecting -1 or +1. Also, its error feedback is expected to make a direct contribution to fast tracking behaviour since it creates a new and meaningful input source in which the conventional neuron models does not, hence it is able to induce fast identification capability.

2.2 Optimal Learning Algorithm

Scalero and Tepedelenlioglu proposed a new algorithm in which the weights are updated by minimizing the mean-square error between the desired net and the actual net of the individual neuron in the output layer. It is still not the optimal method. The main reason is that the desired outputs of the hidden layers are obtained through the same way as backpropagation. For solving those problems addressed above and getting the desired outputs of hidden layers for new neuron model, a nearly optimal learning algorithm is proposed in the paper. In this algorithm, the desired outputs of network corresponding to each inputs are inverted through the activation functions as the desired net to the nonlinearity. The weight matrix (W_L) of output layer and the output vector (x_{L-1}) of the previous layer are set with two independent variables. A pair of optimal solution (W_L^*, x_{L-1}^*) is then obtained by minimizing the total mean-squared error between the desired net and the actual net. Let x_{L-1}^* be the desired output of the last hidden layer.

For a brief expression, a neural network with only are hidden layer is considered. If there are N nodes, P nodes and M nodes in the input, hidden layer and output layer respectively, it is define the energy function of the network as,

$$E = \frac{1}{2}\sum_{k=0}^{M}(d_{Lk} - \mathbf{x}_j^T \mathbf{w}_{Lk})^2 \qquad (7)$$

where $w_{Lk} = [w_{Lk0}, w_{Lk1}, ..., w_{LkP}]^T$ is the weight vector between the hidden and the k-th node of output layer. $\mathbf{x}_j = [x_{j0}, x_{j1}, ..., x_{jP}]^T$ is output of the hidden layer.
Our goal is to find the optimal output x_j^* of the hidden layer under the assumption that the weights were obtained by RLS optimal method. Minimizing E with respect to x_i which be written as Eq.(9) in matrix form,

$$\frac{\partial E}{\partial x_j} = 0 \Rightarrow \sum_{k=0}^{M} d_{Lk} w_{Lk} = \sum_{k=0}^{M} d_{Lk} w_{Lk}^T \quad (8)$$

$$W_L d_L = W_L W_L^T x_j \quad (9)$$

where $W_L = [w_{L1}, ... w_{Lk}, ..., w_{LM}]$ is the weights matrix ($P \times M$) of the output layer.
$\mathbf{d}_L = [d_{L1} ..., d_{Lk}, ..., d_{LM}]^T$ is the desired net vector (M□1) of the output layer. As shown in Eq.(9), it is a problem of P unknown parameters with M equations. There are two conditions to solve $\mathbf{x}_j^* = [x_{j1}^*, ..., x_{jk}^*, ..., x_{jM}^*]$

which are;

① P? M, There exists one or more solution x_j^* satisfying Eq.(9). x_j^* can be obtained by using the minimum norm method.

$$x_j^* = W_L (W_L^T W_L)^{-1} d_L \quad (10)$$

② P<M, There is no exact solution. The least-square approximate method can be used to find the solution.

$$x_j^* = (W_L W_L^T)^{-1} W_L d_L \quad (11)$$

Recall that, $x_j^* = [x_{j1}^*, ..., x_{jk}^*, ..., x_{jM}^*]$, where x_{jk}^* is the output of the k-th neuron in the hidden layer. If the bipolar sigmoid function is chosen as the network's activation function, the desired net of hidden layer is expressed as,

$$d_{jk}^* = f(x_{jk}^*)^{-1} \quad (12)$$

The desired linear value of the hidden layer is obtained by the backward adaptation of Eq.(12) instead of Eq.(2) in the new learning algorithm. The covariance matrix(Eq.(4)) and the gain(Eq.(3)) are obtained through RLS(recursive least squares)method about input vector without the bias since the bias does not affect the weights. The weights except that of the bias input can be updated by Eq.(13).

$$w_{jk} = w_{jk} + k_j (d_{jk}^* - y_{jk} + \mu e_{jk}(t-1)) \quad (12)$$

III. System Identification and Experiment Results

Computer simulation of the system identification has been done for the performance analysis of the new learning algorithm and the error self-recurrent neural networks. As shown in Fig.4, the time delayed plant output and the control input are connected to the input layer for the dynamic system identification using multilayer feedforward neural networks.

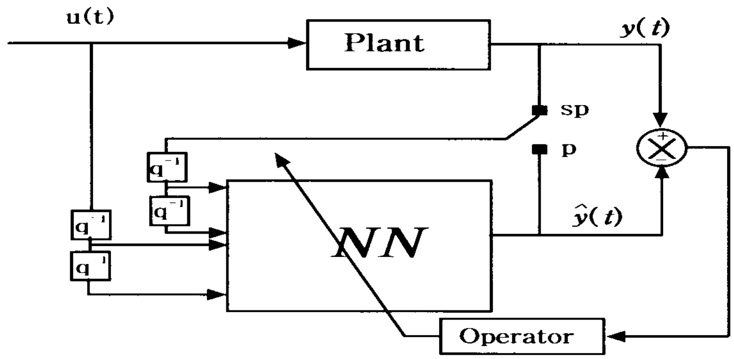

Fig. 4 Identification methods for dynamic systems

For the computer simulation, a complex nonlinear plant model is selected by Eq(15) and input signal is assumed to be

$$u(t) = 0.5\sin(2\pi t/50) + 0.5\sin(2\pi t/120) \quad (14)$$

$$y(t) = (0.85y(t-1)y(t-2) + 0.16u(t-1) + 0.25u(t-2))/(1 + y^2(t-1)) \quad (15)$$

Architecture of neural networks is a two layer feedforward perceptron with five input nodes, seven hidden nodes and one output node. The initial values of weight vector are randomized between 0 and 0.5. The learning rate(μ) is selected to 0.05 for BP and Scalero algorithm. Fig. 5 and 6 show the output response of neural network trained by BP algorithm and Scalero algorithm. The result shows that Scalero's algorithm converges faster than BP, however it converges after 200 steps. Fig. 7 is the output response of the neural network trained by the new learning algorithm of the error self-recurrent. The result shows that the proposed new algorithm converges faster than the Scalero algorithm.

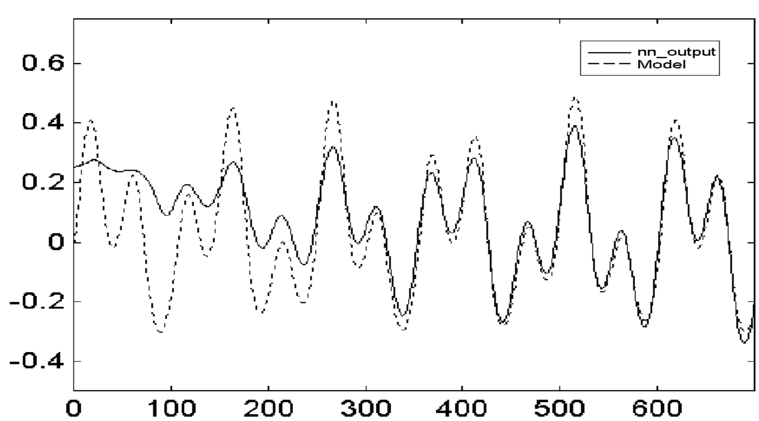

Fig. 5 Network output by BP algorithm(bias value is +1)

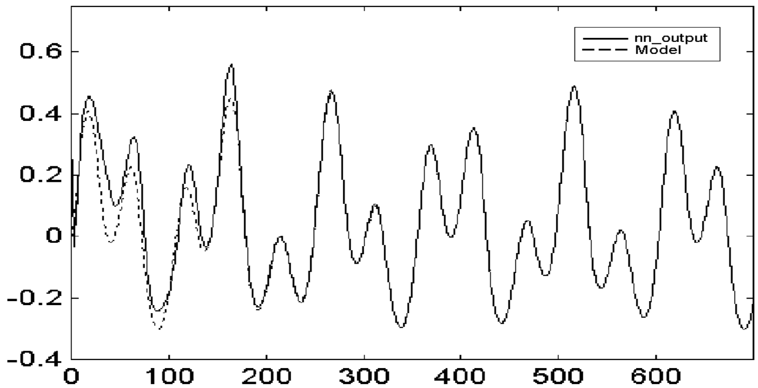

Fig. 6 Network output by Scalero's algorithm(bias value is +1)

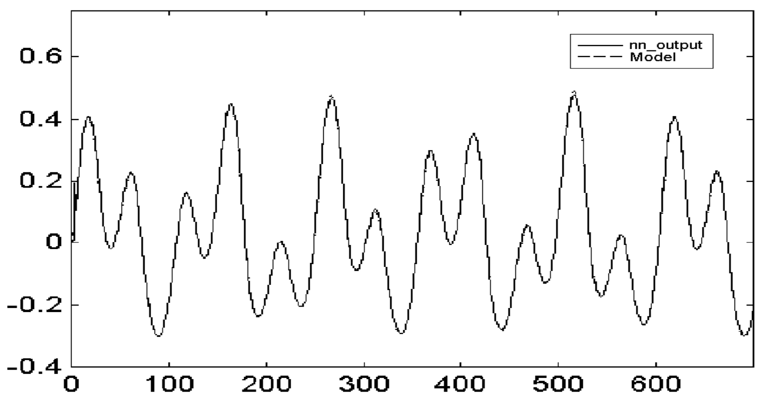

Fig. 7 Network output by proposed algorithm(bias value is $e_{jk}(t-1)$)

IV. Neural Network based Adaptive PID Controller and Experiment Results

We will depict simply on the design of PID controller with neural networks model suggested by Keyser[8] and judge the ability of PID controller based on error recurrent neural networks model. Generally, the one-step ahead nonlinear model of nonlinear by means of neural networks is expressed as,

$$y^{nn}(t+1) = f(w_L^T f(w_{L-1}^T \dots f(w_1^T x_1))) \quad (16)$$

where, $x_1 = [y(t),\dots,y(t-n+1),u(t),\dots,u(t-m)]^T$

PID controller has been widely used in industry. The discrete time PID controller usually has a structure described by:

$$u(t)=u(t-1)+K_1e(t)+K_2e(t-1)+K_3e(t-2) \quad (17)$$

Assume that the neural networks model is sufficiently accurate. In the predictive control system, the predictive output of neural networks , not the plant output is fed directly into the controller Therefore, the signal flowing into the PID controller is the error between the setpoint of system and the predictive output of neural network. $(e(t) = r(t+1) - y^{nn}(t+1))$.

Keyser defines the control performance index as,

$$J = 1/2[e(t)]^2 = 1/2[r(t+1) - y^{nn}(t+1)]^2 \quad (18)$$

The gains of PID controller can be adjusted. It is based on the gradient descent method of the above performance index, i.e.

$$K_i(t) = K_i(t-1) - \lambda \frac{\partial J}{\partial K_i(t)} = K_i(t-1) + \lambda e(t) \frac{\partial y^{nn}(t+1)}{\partial u(t)} \frac{\partial u(t)}{\partial K_i(t)} \quad (19)$$

where λ is a positive factor called optimizing step. The derivative of $y^{nn}(n+1)$ with respect to u(t) can be obtained from the neural network model, and also the control signal can be solved from the neural network model. You may refer to a reference[8] if you need to know the content in detail about the design of PID controller with neural networks model. We select Eq.(20) as servo model of the robot manipulator to examinate control ability, and test the performance of the PID controller on-line based on the model of neural networks and the general PID controller tuned by Ziegler-Nichols law.

$$\begin{aligned} y(t) = {} & 0.2y^2(t-2) + 0.2y(t-1) + 0.25y(t-2) + 0.25u(t-1) \\ & + 0.45\sin(0.5(y(t-1) + y(t-2)))\cos(0.5(y(t-1) + y(t-2))) \end{aligned} \quad (20)$$

As shown in Fig. 8, there still exists the steady-state error. If the parameters of plant change, the ability of adaptation is decreased. Fig. 9 and 10 show the response of the adaptive PID controller based on the neural networks model. At the initial stage of control, the system identification is being processed, so system is not properly controlled. But as time goes on, the system follows the desired response. The proposed controller, the adaptive PID controller based on the ER neural network model, has a fast tracking. Thus it is possible to design the real-time controller based on the neural networks model.

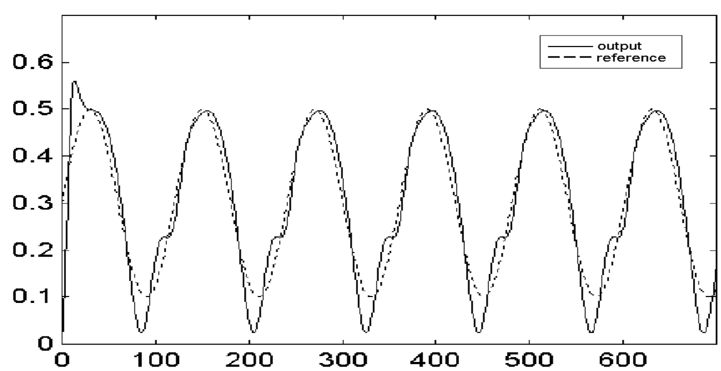

Fig. 8 Control response by conventional PID controller

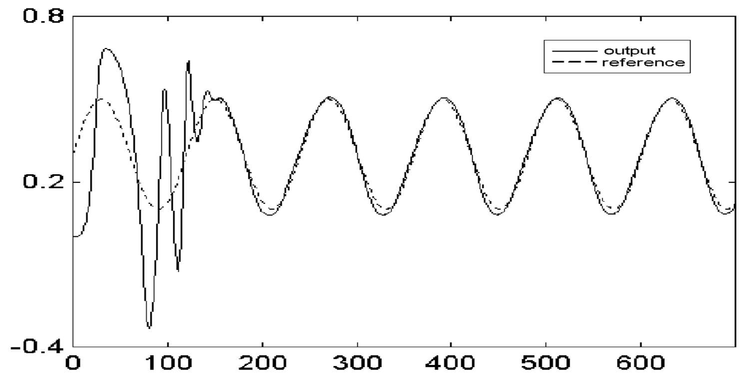

Fig. 9 Control response by BP neural networks based PID controller

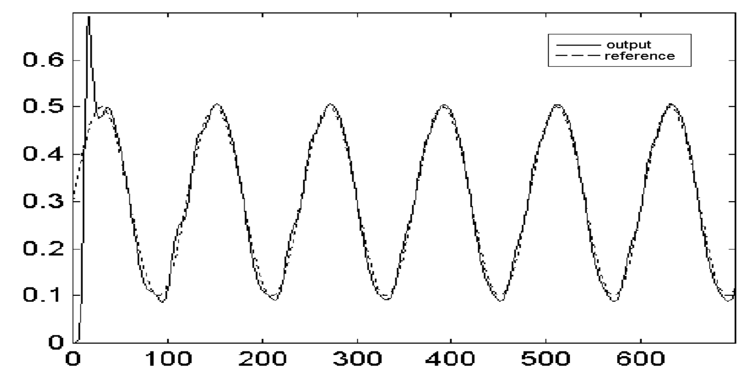

Fig. 10 Control response by ER neural networks based PID controller

V. Conclusion

In this paper, we present a novel error self-recurrent neural model with a bias input, which is the time-delayed error signal between the desired output and the model output instead of +1 or -1. To train the proposed neural networks, we obtain the desired hidden layer value by using the optimal law instead of the backpropagation algorithm, and develop a novel learning algorithm that estimates each weights on-line by RLS method. Experimental studies showed that the proposed method is not influenced from the initial weight values. By resolving the problem of selecting the learning rate, we made it flexible to design the controller in real-time based on neural networks model. In addition, we designed an adaptive PID controller by using the error recurrent neural networks, and also got a good result in the case of robot manipulator experiment. Future research is to prove ER neural model convergence and the controller design based on the multilayer ER neural network model.

REFERENCE

1) Narendra. K. S. and Patthasarathy, "Identification and control of dynamical systems using neural

networks," IEEE Trans. Neural Networks, Vol. 1, pp. 4-27, 1990
2) Wabgaonkar, H. and Stubberud, A., "A system identification technique based on neural networks," Proc. IEEE Int. Conf. Systems Engineering, Kobe, Japan, pp.352-356, Sept. 1992
3) Rumelhart, D. E. and McClelland, J. L., "Parallel Distributed Processing," Vol. 1, MIT Press, Cambridge, MA, 1986
4) Wasserman, P. D., Neural Computing, Theory and Practice, Van Nostrand Reinhold, NY, 1989
5) Singhal, S. and Wu, L., "Traning feedforward networks with the extended Kalman algorithm," Proc. IEEE Int. Conf. Acoustic, Speech, Signal Processing, pp.1187-1190, May 1989
6) Scalero, R. S. and Tepedelenliouglu, N., "A fast new algorithm for traning feedforward neural networks," IEEE Trans. Signal Processing, Vol. 40, pp. 202-210, 1992
7) Haykin, S., Neural Networks : A Comprehensive Foundation, Macmillan College Publishing Company Inc., NY, 1994
8) R. D. Keyser and Yonghong Tan, "Adaptive PID control with neural network based predictor", CONTROL'94, pp.1490-1494, 1994