'94 KACC (1994. 10. 17 ~ 20)

A Learning Control of DC Servomotor Using Neural Network

Hiroaki Kawabata*, Katsuhisa Yamada**,
Zhang Zhong* and Yoji Takeda**

* Department of Communication Engineering, Faculty of Computer Science and System Engineering, Okayama Prefectural University
111, Kuboki, Soja, Okayama, 719-11 Japan
** Department of Electrical and Electronic Systems, College of Engineering, University of Osaka Prefecture
1-1, Gakuen-cho Sakai, Osaka, 593 Japan

Abstract

This paper proposes a method of learning control in DC servomotor using a neural network. First we estimate the pulse transfer function of the servo system with an unknown load, then we determine the best gains of I-PD control system using a neural network. Each time the load changes, its best gains of the I-PD control system is computed by the neural network. And the best gains and its pulse transfer function for the case are stored in the memory. According the increase of the set of gains and its pulse transfer function, the learnig control system can afford the most suitable I-PD gains instantly.

1. Introduction

In a model following servosystem, it is desired for outputs of servomortor to precisely follow the outputs of a given reference model. For a given load in DC servomotor, it is feasible to automatically adjust the gains of the I-PD control system in DC sevomortor using a neural network.[1] In a simple neural network as determination of the I-PD control gains, the coefficients of coupling in the neural network can be itself to be seeked instead the output of the neural network. The neural network for the purpose is rather simple, but the neural network needs to learn the gains in every time the load changes. For the load which is varing all the time or for the load which encounters for the first time, it is necessary to compute the gains using the neural network.

In this paper we first estimate the pulse transfer-function of the servo system with an unknown load, then we determine the best gains of I-PD control system using a neural network. Then, we propose a learning control system in which the desired I-PD gains are determined by using the appropriate gains which is obtained by the linear combination of the set of gains already computed.

2. Learning Control System

Our learning control system is shown in Fig.1.

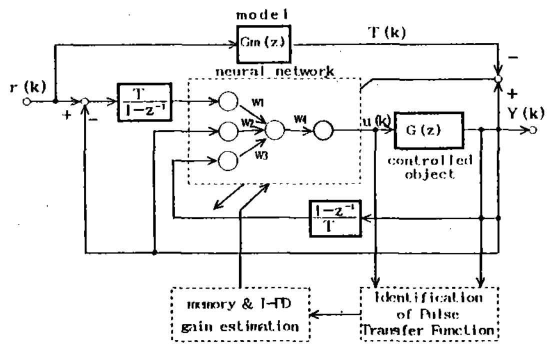

Fig.1 I-PD control system using neural network

The inputs of the identification of pulse transfer function are the set of input and output of controlled object, and the load characteristics of the DC servomotor with a load is identified as the set of coefficients of the pulse transfer function. The set of coefficients of the estimated transfer function is transfered into the memory and I-PD gain estimation. In the neural network the best I-PD gains are learned, and the I-PD gains obtained are stored in the memory and I-PD gain estimation with its coefficients of the pulse transfer function. Although each time the load characteristics of the DC servomotor changes, the best gains of the I-PD control system must be computed by the neural network in the begining, according to increase of the set of memories of the best gains, the memory and I-PD gain estimation gets to have the ability to estimate the best gains for the case by using a linear combination of the set of memories of the best gains already computed. If the initial gains estimated by the linear combination are not suited, the gains are renewed by the learning of the neural network. After these iterations of the learning and estimation, the learning control system can afford the appropriate gains by the estimation using these already obtained gains. When a quite new load is added in the DC servomotor, in general, many iterations are necessary for the learning in neural networks, because random variables are used as the initial values of the neural network. But, in our case, by using the appropriate gains which is obtained by the linear combination as the initial values of the neural network, our neural network rapidly converges to precise values. Through these procedures, such as identification of pulse transfer function, computation of gains by using neural network, memory of the set of pulse transfer function and best gains, an interpolation or an extrapolation as the initial values of the neural network, our learning control system progresses in learning, and it becomes to be able to deal with the best gains imediately for the DC servomotor which encounters a new load.

3. Identification of Pulse Transfer Function

In this section we show the ordinally estimation method of pulse transfer function using the method of least squares.[2] The pulse transfer function is generally described by the following equation.

$$G(z^{-1}) = \frac{Y(z)}{U(z)} = \frac{\beta_1 z^{-1} + \beta_2 z^{-2} + \cdots + \beta_m z^{-m}}{1 - \alpha_1 z^{-1} - \alpha_2 z^{-2} \cdots - \alpha_n z^{-n}} \quad (1)$$

Where we assume the order of n and m are known. The problem is to determine the $n+m$ parameters, $\alpha_1 \sim \alpha_m$ $\beta_1 \sim \beta_n$ when the inputs and outputs time series $u(1) \sim u(N)$ and $y(1) \sim y(N)$ are given. We assume the following vectors $\mathbf{x}$ and $\mathbf{z(k)}$.

$$\mathbf{x} = [\alpha_1, \alpha_2 \cdots \alpha_n, \beta_1, \beta_2 \cdots \beta_m]^T \quad (2)$$

$$\begin{aligned}\mathbf{z(k)} = [&y(k-1), y(k-2), \cdots y(k-n),\\ &u(k-1), u(k-2), \cdots u(k-n)]^T \end{aligned}\quad (3)$$

Since the relation between u(k) and y(k) is given by Eq.4, the relation is described in Eq.5 using $\mathbf{x}$ and $\mathbf{Z}$. Where vector $\mathbf{y}$ and matrix $\mathbf{Z}$ are defined as follows

$$\begin{aligned}y(k) = &\alpha_1 y(k-1) + \alpha_2 y(k-2) + \cdots + \alpha_n y(k-n)\\ &+\beta_1 u(k-1) + \beta_2 u(k-2) + \cdots + \beta_m u(k-n)\end{aligned} \quad (4)$$

$$\mathbf{y} = \mathbf{Zx} \quad (5)$$

$$Z = \begin{bmatrix} z^T(n+1) \\ \vdots \\ z^T(N) \end{bmatrix} \quad (6)$$

$$\mathbf{y} = [y(n+1), y(n+2), \cdots, y(N)]^T \quad (7)$$

Describing the estimation of $\mathbf{x}$ as $\hat{\mathbf{x}}$ we define the error vector $\mathbf{e}$.

$$\mathbf{e} = \begin{bmatrix} e(n+1) \\ \vdots \\ e(N) \end{bmatrix} = \mathbf{y} - \mathbf{Z}\hat{\mathbf{x}} \quad (8)$$

Assuming the cost function J

$$J = \mathbf{e}^T\mathbf{e} = [\mathbf{y} - \mathbf{Z}\hat{\mathbf{x}}]^T[\mathbf{y} - \mathbf{Z}\hat{\mathbf{x}}] \quad (9)$$

the estimation $\hat{\mathbf{x}}$ which minimize J is given

$$\hat{\mathbf{x}} = (\mathbf{Z}^T\mathbf{Z})^{-1} \cdot (\mathbf{Z}^T\mathbf{y}) \quad (10)$$

The online estimation is also given by using the up to date input and output. When the vectors $\hat{\mathbf{x}}_N$, $\mathbf{y}_N$ and $\mathbf{Z}_N$ at time N are given, the new estimation $\hat{\mathbf{x}}_{N+1}$ at time $N+1$ is given by using the vectors $\mathbf{y}_{N+1}$ and $\mathbf{Z}_{N+1}$ as follows

$$\mathbf{y}_{N+1} = \begin{bmatrix} \mathbf{y}_N \\ y(N+1) \end{bmatrix} \quad (11)$$

$$\mathbf{Z}_{N+1} = \begin{bmatrix} \mathbf{Z}_N \\ \mathbf{z}^T(N+1) \end{bmatrix} \quad (12)$$

$$\hat{\mathbf{x}}_{N+1} = \hat{\mathbf{x}}_N + \mathbf{g}_{N+1}\{y(N+1) - \mathbf{z}^T(N+1)\hat{\mathbf{x}}_N\} \quad (13)$$

$$\mathbf{g}(N+1) = \frac{\mathbf{P}_N\mathbf{z}(N+1)}{1 + \mathbf{z}^T(N+1)\mathbf{P}_N\mathbf{z}(N+1)} \quad (14)$$

$$\mathbf{P}_N = [\mathbf{I} - \mathbf{g}(N)\mathbf{z}^T(N)]\mathbf{P}_{N-1} \quad (15)$$

4. Learning Algorithm

Since the transfer function of the DC servomotor under the consideration is described by a linear 2 order system, the neural network may be rather simple.

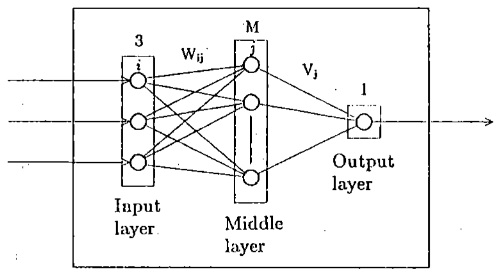

Fig.2 3 inputs neural network

Fig.(2) shows the neural network which is composed by 3 input layer, M hidden layer and a output layer. The output of j-th unit in hidden layer h_j is given by using the outputs of input layer I_i

$$h_j = \sum_{i=1}^{3} w_{ij} I_i \quad (16)$$

Similarly the output of the output layer O is given:

$$O = \sum_{i=1}^{M} v_j h_j \quad (17)$$

The error function E_p is assumed by using the difference between the output of the controlled object Y_k and the teacher signal T_k.

$$E_p = \frac{1}{2}(T_k - Y_k)^2 \tag{18}$$

The derivative E_p for the variance of the coupling parameter v_j is given as follows:

$$\frac{\partial E_p}{\partial v_j} = \frac{\partial E_p}{\partial Y_k}\frac{\partial Y_k}{\partial O_{k-1}}\frac{\partial O_{k-1}}{\partial v_j} \tag{19}$$

$$\frac{\partial E_p}{\partial Y_k} = -(T_k - Y_k) \tag{20}$$

$$\frac{\partial O_{k-1}}{\partial v_j} = h_j \tag{21}$$

where the suffix k in Y_k and O_{k-1} shows the sampling time k. Therefore the variance Δv_j is given as follows.

$$\Delta v_j = -\alpha_1\left(\frac{\partial E_p}{\partial v_j}\right) = \alpha_2 (T_k - Y_k)\, h_j \tag{22}$$

where α_1 and α_2 are the learning constants. Similarly the derivative E_p for the variance of the connection parameter w_{ij} is given as follows:

$$\frac{\partial E_p}{\partial w_{ij}} = \frac{\partial E_p}{\partial h_j}\frac{\partial h_j}{\partial w_{ij}} = \frac{\partial E_p}{\partial Y_k}\frac{\partial Y_k}{\partial O_{k-1}}\frac{\partial O_{k-1}}{\partial h_j}\frac{\partial h_j}{\partial w_{ij}} \tag{23}$$

$$\frac{\partial E_p}{\partial Y_k} = -(T_k - Y_k) \tag{24}$$

$$\frac{\partial O_{k-1}}{\partial h_j} = v_j \tag{25}$$

$$\frac{\partial h_j}{\partial w_{ij}} = I_i \tag{26}$$

Hence the variance Δw_{ij} is given as follows.

$$\Delta w_{ij} = -\beta_1\left(\frac{\partial E_p}{\partial w_{ij}}\right) = \beta_2 (T_k - Y_k)\, v_j I_i \tag{27}$$

where β_1 and β_2 are the learning constants.

5. Linear Combination of I-PD gains

Each time the new parameters of load characteristics are estimated, the best control gains are learned by the neural network, and the set of the new parameters $\mathbf{x}_i$ and the best gains $\mathbf{w}_i$ is memorized.

$$(\mathbf{x}_i, \mathbf{w}_i) \tag{28}$$

According to increase of memories the learning of the system advances. When the new parameters $\mathbf{x}_n$ are estimated, as the candidate of the best gains, that is, the initial values of the learning $\mathbf{w}_n$ may be given by the linear combination with the neighbor ones which are learned previously as the best gains. We choose the r neighbor sets of the new parameter vector $\mathbf{x}_n$, and make the following Matrix.

$$\mathbf{X} = \left[\ \mathbf{x}_1,\ \mathbf{x}_2,\ \cdots\ \mathbf{x}_r\ \right] \tag{29}$$

Introducing the coefficient vector $\mathbf{k}$, we assume the next relation.

$$\mathbf{k} = \left[\ k_1,\ k_2,\ \cdots\ k_r\ \right]^T \tag{30}$$

$$\mathbf{x}_n = \left[\ \mathbf{x}_1,\ \mathbf{x}_2,\ \cdots\ \mathbf{x}_r\ \right]\begin{bmatrix} k_1 \\ \vdots \\ k_r \end{bmatrix} \tag{31}$$

$$\mathbf{k} = \left[\ \mathbf{x}_1,\ \mathbf{x}_2,\ \cdots\ \mathbf{x}_r\ \right]^{-1}\mathbf{x}_n \tag{32}$$

Hence we can obtain the following linear combination relation.

$$\mathbf{w}_n = \mathbf{W}^T\mathbf{k} \tag{33}$$

where

$$\mathbf{W} = \left[\ \mathbf{w}_1,\ \mathbf{w}_2,\ \cdots\ \mathbf{w}_r\ \right] \tag{34}$$

According to increase of the best gain memories, the above estimation gives a fair performance of the system without the learning. If the control performance is not sufficient, the learning by the neural network is made, and the best gains are found, and memorized. Usually, in order to have a unique solution for Eq.32 the matrix $\mathbf{X}$ is chosen a square matrix. When the matrix $\mathbf{X}$ is not able to be chosen as a square matrix, we can get the following estimation using the singular value decomposition of the matrix $\mathbf{X}$.

$$\mathbf{X} = \left[\ U\ \right]^T\left[\ \lambda\ \right]^{1/2}\left[\ V\ \right] \tag{35}$$

Where [λ] is an eigen value matrix, and [U], [V] are the orthonormal matrices. Then, $\mathbf{k}$ is given by the following equation.

$$\mathbf{k} = [V]^T[\lambda]^{-1/2}[U]\mathbf{x}_n \tag{36}$$

We can obtain $\mathbf{w}_n$ by inserting Eq.36 into Eq.33.

6. Example System and Simulation

We consider the DC servomotor system shown in Fig.3. The system equations are given by the following differential equations.

$$J\frac{d^2\theta_1}{dt^2} = Bli \tag{37}$$

$$\left(J = J_M + \left(\frac{N_1}{N_2}\right)^2 J_L\right)$$

$$Ke = Ri + Bl\frac{d\theta_1}{dt} \tag{38}$$

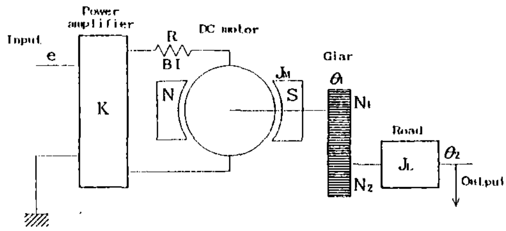

Fig.3 DC servomortor system

Letting the output θ_1 and the input voltage e, we obtain the open loop transfer function $G(s)$.

$$G(s) = \frac{\frac{BlK}{RJ}}{s(s+\frac{B^2l^2}{RJ})} = \frac{k}{s(s+\omega)} \tag{39}$$

We can describe the two order system Eq.39 as the next pulse transfer function.

$$G(z) = \frac{b_1 z + b_2}{z^2 + a_1 z + a_2} \tag{40}$$

We set the transfer function of the reference model.

$$G_m(s) = \frac{\omega_1 \omega_2^2}{(s+\omega_1)(s^2 + 2\zeta\omega_2 s + \omega_2^2)} \tag{41}$$

In numerical simulations, we set $\omega_1 = \omega_2 = 10$ in Eq.41. For the convenience we set the number of hidden layer m=1. We let the neural network learn the best gains for the pulse transfer functions of the four different loads.(case (a)~(d))

The time responses of output for the controlled object are shown in Figs.(4)(a)~(d).

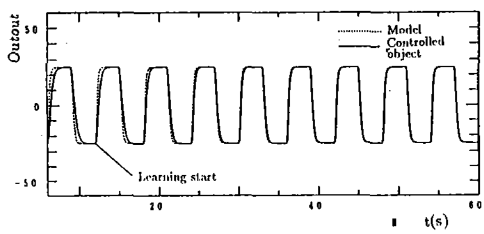

Fig.4 (a) $(a1, a2, b1, b2) =$
$(1.78, -0.779, 2.07\times10^{-3}, 1.91\times10^{-3})$

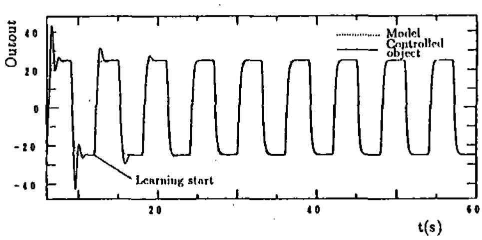

Fig.4 (b) $(a1, a2, b1, b2) =$
$(1.47, -0.472, 1.78\times10^{-3}, 1.39\times10^{-3})$

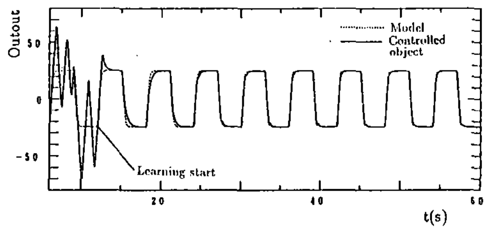

Fig.4 (c) $(a1, a2, b1, b2) =$
$(1.61, -0.607, 9.59\times10^{-4}, 8.12\times10^{-4})$

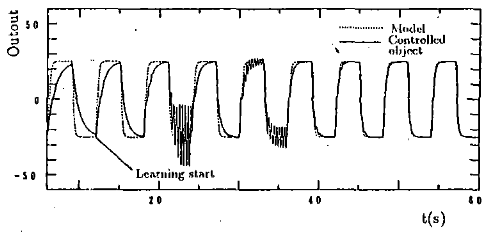

Fig.4 (d) $(a1, a2, b1, b2) =$
$(1.61, -0.607, 2.88\times10^{-3}, 2.44\times10^{-3})$

It is seen that the learning of the neural network works well in every case. The control gains obtained by the learning of neural network and the optimal gains by theoretical calculation are presented in Table.1.

Controlled object		I gain W_1*V	P gain W_2*V	D gain W_3*v
(a)	N. N.	21.2	-6.46	-0.112
	Optimal	22.2	-6.67	-0.111
(b)	N. N.	29.4	-8.95	0.743
	Optimal	22.2	-6.67	1.00
(c)	N. N.	75.8	-23.0	0.166
	Optimal	44.4	-13.3	0.889
(d)	N. N.	16.8	-5.14	0.211
	Optimal	14.8	-4.44	0.296

Table.1. Comparison of the gains obtained by N.N. and the optimal gains

Table.1 indicates that the set of control gains fairly have the robustness. The result tells that we don't need to take so many learning times.

The comparison of the time response of the output using the control gains obtained after 10,000 learning times and the time response of the model system is shown in Fig.5(a)~(d).

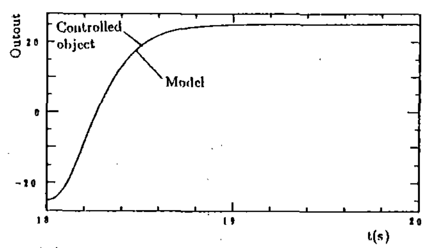

Fig.5 (a) $(a1, a2, b1, b2) =$
$(1.78, -0.779, 2.07\times10^{-3}, 1.91\times10^{-3})$

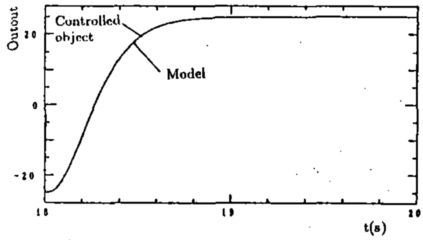

Fig.5 (b) $(a1, a2, b1, b2) =$
$(1.47, -0.472, 1.78 \times 10^{-3}, 1.39 \times 10^{-3})$

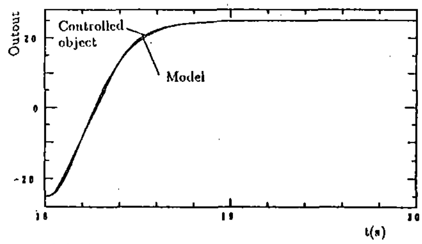

Fig.5 (c) $(a1, a2, b1, b2) =$
$(1.61, -0.607, 9.59 \times 10^{-4}, 8.12 \times 10^{-4})$

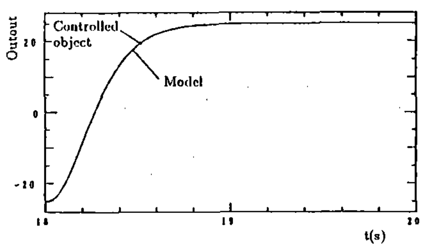

Fig.5 (d) $(a1, a2, b1, b2) =$
$(1.61, -0.607, 2.88 \times 10^{-3}, 2.44 \times 10^{-3})$

Next, we show the result using I-PD gains which is obtained by the linear combination in Fig.6.

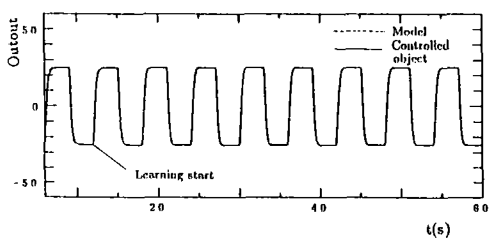

Fig.6 The output response using I-PD gains obtained by the linear combination

It is observed that the result using the gains estimated by the linear combination of the previous gains shows good performance.

7. Conclusion

In this paper we have proposed a method of learning control in DC servomotor using a neural network. Although it is seen clear by intuitive consideration that according to the increase of the set of gains and its pulse transfer function, the learnig control system can afford the most suitable I-PD gains instantly, there are some problems in application. Since the robustness of I-PD gains exist, as is seen in Table.1, the linearity does not always exist among the set of gains which is learned by using neural network. However, sooner or later, since the set of the best gains increases around the unknown load, the linearlity of the linear combination of the gains also has been elevated.

It is difficult problem to estimate the pulse transfer function whose characteristics always change, and it is also difficult to control the object which continuously changes. These are future problems.

References

1) K.Endo, Y.Ishida and T.Honda: "Gain Adjustment of I-PD Control System Using Neural Network", T.IEE Japan, Vol.113-C, No.6, 409 (1993)
2) K. Kanai,"Robust Adaptive Control", Ohmu, 49 (1989)
3) T.Yamada and T. Yabuta: "Neural Network Controller Using Autotuning Method for Nonlinear Functions", IEEE Treans. Neural Networks, 3, 595 (1992)
4) F. Chen: "Back-Propagation Neural Networks for Nonlinear Self-Tuning Adaptive Contreol", IEEE Control Systems Magazine, 10, No.3, 44 (1990)