Proceedings of the

1996 IEEE/RSJ International Conference on

Intelligent Robots and Systems

IROS 96

Robotic Intelligence Interacting with Dynamic Worlds

November 4-8, 1996

Senri Life Science Center, Osaka, Japan

Volume 1

Sponsored by

IEEE Industrial Electronics Society

IEEE Robotics and Automation Society

Robotics Society of Japan

Society of Instrument and Control Engineers

New Technology Foundation

Adaptive Neural Regulator and Its Application to Torque Control of a Flexible Beam

Bing Hong XU, Toshio TSUJI and Makoto KANEKO
Faculty of Engineering, Hiroshima University
Higashi-Hiroshima, 739 JAPAN

Abstract

This paper proposes an adaptive regulator using neural network. For a controlled object with linear and nonlinear uncertainties, the conventional optimal regulator is designed based on a known linear part of the controlled object and the uncertainties included in the controlled object are identified using the neural network. At the same time, the neural network adaptively compensates a control input from the predesigned optimal regulator. In this paper, first, we show how the output of the neural network compensates the control input based on the Riccati equation, and a sufficient condition of the local asymptotic stability is derived using the Lyapunov stability technique. Then, the proposed regulator is applied to the torque control of a flexible beam. Experimental results under the proposed regulator are compared with the conventional optimal regulator in order to illustrate the effectiveness and applicability of the proposed method.

1 Introduction

The optimal regulator is usually designed for a mathematical model of a controlled object. However, the mathematical model is not exactly known in practical application of the optimal regulator. For the controlled object with linear uncertainties, a research of a robust optimal regulator has been conducted [1]. It should be noted that the robust optimal regulator cannot work well if the nonlinear uncertainties of the controlled object exists.

For this problem, various regulators using neural networks have been proposed in recent years. Yamada and Yabuta [2] proposed a learning controller based on a direct neural control approach and used the cost function of the conventional optimal regulation for the neural network training. Also, they discussed the stability of the linear discrete-time SISO (Single Input-Single Output) controlled plant [3]. Although this type of the controller is very simple and can be applied to various feedback control system, the uncertainty of the controlled plant cannot be identified and some parameters included in the neural network is quite difficult to be set. On the other hand, Takahashi [4] used an adaptive neural identifier and the direct neural controller [2] for controlling a flexible arm. The neural identifier can identify the parameters of the arm and the neural controller can work on the basis of the identified parameters. Polycarpou and Helmicki [5] presented a learning approach for automated fault detection and accommodation. Their system can detect plant's fault and can adjust the plant behavior using multiple neural networks. Rovithakis and Christodoulou [6] linearized an unknown nonlinear dynamic system and used three neural networks for making direct adaptive regulator. In these proposed methods, the unknown part of the controlled object is identified by one neural network, and other neural network are used as the compensator. Since the multiple neural networks must be trained, it requires a long time for computation and learning, and the stability analysis becomes difficult.

Generally, the controlled object includes a known part and an unknown part, and the unknown part can deal with linear and nonlinear uncertainties. So, it is necessary that efficient regulator design utilizes such information of the controlled object.

In this paper, an adaptive regulator using a neural network for a class of dynamic system with uncertainties is proposed. The proposed method designs an optimal regulator for the linear known part and uses the neural network to identify the unknown part included in the controlled object. At the same time, the neural network works as an adaptive compensator for the unknown part of the controlled object. In this paper, we show how the output of the neural network compensates the control input based on the Riccati equation and derive a sufficient condition of the local asymptotic stability using the Lyapunov stability technique. Then the proposed regulator is applied to the torque control of a flexible beam. Experimental results under the proposed regulator are compared with the conventional optimal regulator in order to illustrate the effectiveness and applicability of the proposed regulator.

2 Adaptive Regulator Using Neural Network

2.1 Problem Formulation

As a controlled object, we consider the following system that consists of linear and nonlinear part:

$$\dot{x}(t) = A_L x(t) + \tilde{A}(x(t)) + Bu(t), \quad (1)$$

$$y(t) = Cx(t), \quad (2)$$

where $x(t) \in \Re^{n\times 1}$, $u(t) \in \Re^{m\times 1}$ and $y(t) \in \Re^{l\times 1}$ are the state, the input and the output, respectively; $A_L \in \Re^{n\times n}$, $B \in \Re^{n\times m}$, $C \in \Re^{l\times n}$ are the parame-

Proc. IROS 96
0-7803-3213-X/96/ $ 5.00 © 1996 IEEE

ter matrices; and $\tilde{A}(\cdot)$ is the nonlinear function of the state $x(t)$.

The goal of the control is to find the optimal input that minimizes the quadratic performance index of the form

$$J = \int_0^\infty [x^T(t)Qx(t) + u^T(t)Ru(t)]dt, \qquad (3)$$

where $Q \in \Re^{n\times n} \geq 0$, $R \in \Re^{m\times m} > 0$ are the weight matrices specified by the designer. The nonlinear regulator problem for the system (1) is very difficult and is generally solved using one of the linearized techniques that synthesizes an observer and a linear optimal regulator. However, there is many control problems where (1) cannot be linearized appropriately, so that the nonlinear compensation is particularly required. In this paper, we propose the adaptive regulator using a neural network with excellent capabilities of nonlinear mapping, learning ability and parallel computations.

2.2 Optimal Regulator for the Linearized System

First, we divide the matrix $A_L = A_{L_n} + \Delta_{A_L}$ of (1) into the known parameter matrix $A_{Ln} \in \Re^{n\times n}$ and the uncertainties matrix $\Delta_{A_L} \in \Re^{n\times n}$, and assume that the nonlinear function $\tilde{A}(x(t))$ is approximately described as

$$\tilde{A}(x(t)) \approx A^* x(t) + \Delta_{A^*} x(t) \qquad (4)$$

near the operating point of the controlled system. Here, $A^* \in \Re^{n\times n}$ represents the linearized parameter and $\Delta_{A^*} \in \Re^{n\times n}$ represents the unknown linearized modelling error.

Then the system (1) becomes

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t), & (5)\\ A &= A_n + \Delta_A, & (6)\\ A_n &= A_{Ln} + A^*, & (7) \end{aligned}$$

where $\Delta_A = \Delta_{A_L} + \Delta_{A^*}$. Δ_A and $A_n \in \Re^{n\times n}$ are respectively the system uncertainty and the nominal parameter. Also A_{Ln} and $\Delta_{A_L} \in \Re^{n\times n}$ are the known part of the parameter A_L and the parameter uncertainty. The system (5) is assumed to be controllable and observable. The optimal control input $u^*(t)$ that minimizes the performance index J of (3) is given as

$$u^*(t) = -R^{-1}B^T P x(t), \qquad (8)$$

where $P \in \Re^{n\times n}$ is the unique solution of the Riccati equation [7]

$$PA + A^T P - PBR^{-1}B^T P + Q = 0. \qquad (9)$$

Here the solution P is assumed as

$$P = P_n + \Delta_P, \qquad (10)$$

where $P_n \in \Re^{n\times n}$, $\Delta_P \in \Re^{n\times n}$ are respectively the solution for the known linear part of the system (5) and the compensatory solution of the Riccati equation.

Substituting (6), (10) into (9), we have

$$\begin{aligned} &(P_n + \Delta_P)(A_n + \Delta_A) + (A_n + \Delta_A)^T(P_n + \Delta_P)\\ &-(P_n + \Delta_P)BR^{-1}B^T(P_n + \Delta_P) + Q = 0. \qquad (11) \end{aligned}$$

If the quadratic uncertainties, $\Delta_P\Delta_A$, $\Delta_A^T\Delta_P$, $\Delta_P\Delta_P$ are sufficiently small, (11) can be divided into the following two equations:

$$\begin{aligned} &P_nA_n + A_n^TP_n - P_nBR^{-1}B^TP_n + Q = 0, \qquad (12)\\ &\Delta_PA_n + P_n\Delta_A + \Delta_A^TP_n + A_n^T\Delta_P\\ &-\Delta_PBR^{-1}B^TP_n - P_nBR^{-1}B^T\Delta_P = 0. \qquad (13) \end{aligned}$$

Then, substituting the solution P_n of (12) into (13), we can approximately have

$$\Delta_P = \Theta\Delta_A, \qquad (14)$$

where $\Theta \in \Re^{n\times n}$ represents the transformation matrix (See appendix A).

In order to compute the optimal control input $u^*(t)$ of (8), the matrix P is necessary. Although P_n can be obtained from (12), the uncertainty Δ_A is unknown and we cannot determine the compensatory solution Δ_P. Therefore, the neural network is introduced for solving the problem.

2.3 Proposed Regulator Scheme

Figure 1 shows the scheme of the adaptive neural regulator. The identification system shown by the broken line in Fig. 1 is described as

$$\dot{\hat{x}}(t) = A_nx(t) + Bu(t) + x_{NN}(t), \qquad (15)$$

where $\hat{x}(t) \in \Re^{n\times 1}$ and $x_{NN}(t) \in \Re^{n\times 1}$ are the predicted state of the identification system and the output of the neural network, respectively.

Substituting (10), (14) into (8), we have the optimal control input $u^*(t)$ as

$$\begin{aligned} u^*(t) &= -K_nx(t) - \Delta_K(\Delta_Ax(t)), & (16)\\ K_n &= R^{-1}B^TP_n, & (17)\\ \Delta_K &= R^{-1}B^T\Theta, & (18) \end{aligned}$$

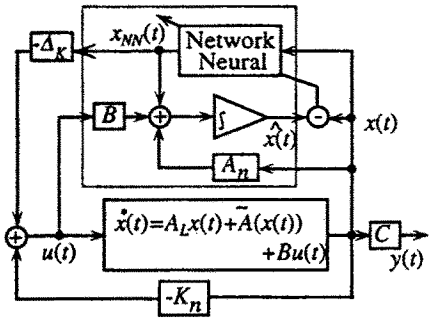

Figure 1: Block diagram of the adaptive neural regulator

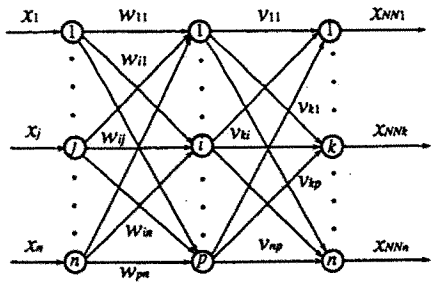

Figure 2: Three-layer neural network

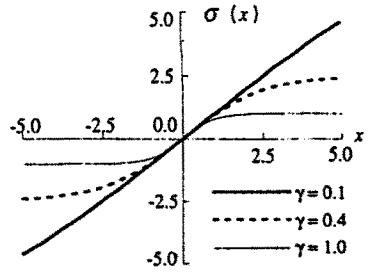

Figure 3: Sigmoid function used neural network

where $K_n \in \Re^{m\times n}$, $\Delta_K \in \Re^{m\times n}$ are respectively the feedback gain and the compensatory gain. The identified state error $\epsilon(t) \in \Re^{n\times 1}$ between the predicted state $\hat{x}(t)$ and the state $x(t)$ is defined as

$$\epsilon(t) = \hat{x}(t) - x(t). \tag{19}$$

Then, the neural network is trained by minimizing the energy function $E(t) = 1/2[\epsilon^T(t)\epsilon(t)]$. If $E(t)$ becomes zero, the predicted state $\hat{x}(t)$ agrees with the state $x(t)$. In other words, the output $x_{NN}(t)$ of the neural network agrees with the state uncertainties $\Delta_x(t) = \Delta_A x(t)$, that is,

$$x_{NN}(t) = \Delta_x(t). \tag{20}$$

Substituting (20) into (16), we can get the optimal input rewritten as

$$u^*(t) = -K_n x(t) - \Delta_K x_{NN}(t). \tag{21}$$

After training the neural network, the control input shown in Fig. 1 must be close to the optimal control input.

2.4 Neural Network and Learning Algorithm

A multi-layer neural network used in the proposed regulator is shown in Fig. 2. The numbers of the units of the input layer, the hidden layer and the output layer are n, p and n, respectively. In Fig. 2, w_{ij} represents the weight that connects the unit j of the input layer to the unit i of the hidden layer; v_{ki} represents the weight that connects the unit i of the hidden layer to the unit k of the output layer. The weight matrices are represented as $W(t) \in \Re^{p\times n}$ and $V(t) \in \Re^{n\times p}$, respectively. Also, the input and output vectors of the neural network are represented as $x(t)$, $x_{NN}(t)$, respectively.

Let the unit j's output of the input layer be $I_j = x_j(t)$ $(j = 1, \cdots, n)$, the unit i's output of the hidden layer be $H_i = \sigma(s_i)$, $s_i = \sum_{j=1}^{n} w_{ij} I_j$, where the sigmoid function $\sigma(\cdot)$ is defined as $\sigma(\mu) \equiv \frac{1}{\gamma} tanh(\gamma\mu)$. Here, γ is the positive parameter related with the shape of the sigmoid function. Fig. 3 shows the input-output relation of the sigmoid function. When $\gamma < 0.1$, the $\sigma(\mu)$ can be approximated by the linear function, and when $\gamma \geq 1$, the $\sigma(\mu)$ takes the form of the *tanh* function. Also, let the unit k's output of the output layer be $O_k = \sigma(\kappa_k)$, $\kappa_k = \sum_{i=1}^{p} v_{ki} H_i$.

For the neural network's training process, the energy function $E(t)$ is rewritten as

$$E(t) = \frac{1}{2}\sum_{k=1}^{n}[\hat{x}_k(t) - x_k(t)]^2 = \frac{1}{2}\sum_{k=1}^{n}\epsilon_k^2(t). \tag{22}$$

In the training process, the energy function is minimized by changing the weights w_{ij} and v_{ki}. According to the back-propagation algorithm [8], the weight updating rules can be described as

$$v_{kj}(t+\Delta t) = v_{kj}(t) - \eta\frac{\partial E(t)}{\partial v_{kj}(t)}, \tag{23}$$

$$w_{ij}(t+\Delta t) = w_{ij}(t) - \eta\frac{\partial E(t)}{\partial w_{ij}(t)}, \tag{24}$$

where $\eta > 0$ is the learning rate, Δt is the time interval of the network learning.

By using (15), (19), (21) and (22), $\partial E(t)/\partial v_{ki}(t)$ can be written as

$$\begin{aligned}\frac{\partial E(t)}{\partial v_{ki}(t)} &= \sum_{q=1}^{n}\epsilon_q(t)\frac{\partial \hat{x}_q(t)}{\partial v_{ki}(t)} \\ &= \sum_{q=1}^{n}\epsilon_q(t)\frac{\partial}{\partial v_{ki}(t)}\{\int_0^t x_{NN_q}(\tau)d\tau \\ &\quad -\sum_{h=1}^{m} b_{qh}\int_0^t\sum_{l=1}^{n}\Delta_{K_{hl}}x_{NN_l}(\tau)d\tau\} \\ &= \sum_{q=1}^{n}\sum_{s=1}^{n}\zeta_{qs}\epsilon_q(t)\frac{\partial x_{NN_s}(t)}{\partial v_{ki}(t)},\end{aligned} \tag{25}$$

$$\zeta_{qs} = \begin{cases} (1-\sum_{h=1}^{m} b_{qh}\Delta_{K_{hq}})\frac{\partial S_q(t)}{\partial x_{NN_q}(t)} & (q=s) \\ -\sum_{h=1}^{m} b_{qh}\Delta_{K_{hq}}\frac{\partial S_s(t)}{\partial x_{NN_s}(t)} & (q\neq s), \end{cases} \tag{26}$$

$$S_i(t) = \int_0^t x_{NN_i}(\tau)d\tau, \tag{27}$$

where a_{qr}, k_{hr}, b_{qh}, $\Delta_{K_{hl}}$ are the elements of the matrices A_n, K_n, B, Δ_K, respectively. The partial

derivative $\partial S_i(t)/\partial x_{NN_i}(t)$ can be approximated as

$$\frac{\partial S_i(t)}{\partial x_{NN_i}(t)} \approx \frac{\Delta S_i(t)}{\Delta x_{NN_i}(t)}. \quad (28)$$

If the $x_{NN_i}(t)$ is only changed by the difference $\Delta x_{NN_i}(t)$, the variation $\Delta S_i(t)$ of $S_i(t)$ becomes

$$\begin{aligned}\Delta S_i(t) &\approx [\sum_{j=0}^{N_t} x_{NN_i}(j\Delta t_s)\Delta t_s + \Delta x_{NN_i}(t)\Delta t_s] \\ &\quad -\sum_{j=0}^{N_t} x_{NN_i}(j\Delta t_s)\Delta t_s, \\ &= \Delta x_{NN_i}(t)\Delta t_s. \quad (29)\end{aligned}$$

Therefore, we can approximate $\partial S_i(t)/\partial x_{NN_i}(t)$ as follows:

$$\frac{\partial S_i(t)}{\partial x_{NN_i}(t)} \approx \Delta t_s, \quad (30)$$

where Δt_s is the small sampling time and $t = N_t\Delta t_s$. Substituting (30) into (26) gives

$$\zeta_{qs} \approx \begin{cases} (1-\sum_{h=1}^{m} b_{qh}\Delta_{K_{hq}})\Delta t_s & (q=s) \\ -\sum_{h=1}^{m} b_{qh}\Delta_{K_{hq}}\Delta t_s & (q\neq s). \end{cases} \quad (31)$$

From (23), we have

$$v_{ki}(t+\Delta t) = v_{ki}(t) - \eta\sum_{q=1}^{n}\sum_{s=1}^{n}\zeta_{qs}\epsilon_q(t)\frac{\partial x_{NN_s}(t)}{\partial v_{ki}(t)}. \quad (32)$$

Also, the updating rule (24) reduces to the following form

$$w_{ij}(t+\Delta t) = w_{ij}(t) - \eta\sum_{q=1}^{n}\sum_{s=1}^{n}\zeta_{qs}\epsilon_q(t)\frac{\partial x_{NN_s}(t)}{\partial w_{ij}(t)}. \quad (33)$$

2.5 Stability Analysis

This section deals with the local asymptotic stability for the system (5) near the optimal set of the weights of the neural network. If the multi-layer neural network is used, there exists the optimal set of the weights that makes the identified error $\epsilon(t)$ zero [9]. Near the optimal set of the weights, the neural network's output $x_{NN}(t)$ can be approximately linearized as

$$x_{NN}(t) \approx \varrho V^T(t)W(t)x(t), \quad (34)$$

where $\varrho > 0$ represents the gradient of the sigmoid function.

From (5), (15) and (34), the identified error (19) is described by

$$\begin{aligned}\epsilon(t) &= \int_0^t [A_n x(\tau) + Bu(\tau) + x_{NN}(\tau) - A_n x(\tau) \\ &\quad -Bu(\tau) - \Delta_A x(\tau)]d\tau \\ &= \Phi(t)\int_0^t x(\tau)d\tau, \quad (35) \\ \Phi(t) &= [\varrho V^T(t)W(t) - \Delta_A)], \quad (36)\end{aligned}$$

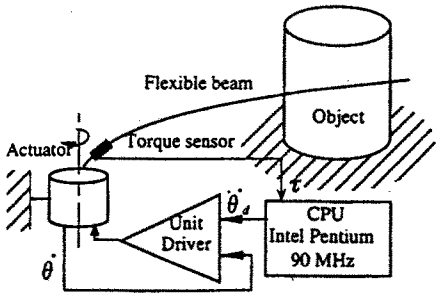

Figure 4: Experimental setup for torque control of a flexible beam

where $\Phi(t) \in \Re^{n\times n}$ is defined as the parameter error.

According to Fig. 1, if the identified error $\epsilon(t)$ can be asymptotically stabilized, the asymptotic stability of the proposed regulator can also be guaranteed. Since the controlled system is assumed to be controllable, the system state $x(t)$ is bounded. In order to assure stability of the identified error, the stability of the parameter error $\Phi(t)$ should be guaranteed.

We consider a Lyapunov function $\Psi(t)$ of the following form

$$\Psi(t) = [\mathrm{cs}\Phi(t)]^T\mathrm{cs}\Phi(t), \quad (37)$$

where $\mathrm{cs}\Phi(t)$ is the expanded form of the column of the matrix $\Phi(t)$. The difference Δ_Ψ of the Lyapunov function $\Psi(t)$ is defined as

$$\Delta_\Psi = \Psi(t+\Delta t) - \Psi(t). \quad (38)$$

If $\Delta_\Psi < 0$, the asymptotic stability of the proposed regulator can be guaranteed by the Lyapunov's stability method.

When the neural network is trained sufficiently until the identified error becomes zero, the sufficient condition of the local asymptotic stability is that the learning rate η is chosen as

$$\rho_1/(1+\rho_2) > \eta > 0, \quad (39)$$

where ρ_1, ρ_2 are the robust margins depended on the weights of the neural network (Proof: see appendix B).

Since the small learning rate η satisfying the condition (39) can be chosen easily, the stability of the proposed regulator can be also guaranteed.

3 Application to Torque Control of a flexible Beam

In this section, our goal is to control the joint torque according to the reference torque of the flexible beam contacted with the object as shown in the Fig. 4. The contact point between the flexible beam and the object can be detected by active motion of the joint [10]. So, if the joint torque can be controlled, the force applied to the object may also be controlled. However, the characteristics of the flexible beam are nonlinearly changed depending on material and shape of the beam,

a contact force, a frictional force and so on. Also, since the beam's stiffness of the joint is varied with the distance between the joint and the contact point on the object, it is very difficult to obtain the exact dynamic model of the flexible beam [10] . In this section, the Adaptive Neural Regulator (ANR) is applied to the torque control of the flexible beam in order to illustrate the effectiveness and applicability of the ANR.

3.1 Experimental Device

An experimental device for the torque control is shown in Fig. 4 [11]. The beam is steel, 0.32 m in length, and 0.5 mm in diameter. The torque sensor is made of a semiconductor gauge put on an aluminum sheet. When the beam contacts with a fixed object, the torque τ of the beam joint can be measured by the torque sensor. The actuator is velocity-controlled with the reference angular velocity of the beam. It should be noted that a driving torque of the actuator cannot be controlled directly. The experimental device includes various nonlinear and unknown uncertainties.

3.2 Formulation

First, the reference angular velocity $\dot{\theta}_d$ can be considered as the input to the flexible beam, so the transfer function from $\dot{\theta}_d$ to the measured torque τ of the joint can be approximately described by [11]

$$H_n(s) = \frac{K_s K_b}{s(T_f s + 1)} = \frac{b_0}{s^2 + a_1 s}, \qquad (40)$$

where $b_0 = (K_s K_b)/T_f$, $a_1 = 1/T_f$. K_s is the gain, K_b is the elastic constant of the beam, and T_f is the time constant of the velocity controlled system.

Here, let the rectangular reference value be denoted as $r(t)$. The control error is defined as $e(t) = r(t) - \tau(t)$. From (40) we have the following state equation

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -a_1 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t), \qquad (41)$$

$$y(t) = [b_0 \quad 0] \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}, \qquad (42)$$

where $x_1(t) = e(t)$, $x_2(t) = \dot{e}(t)$. Then the quadratic performance index is defined as

$$J = \int_0^{t_f} [x^T(t) Q x(t) + u^T(t) R u(t)] dt, \qquad (43)$$

where $t_f \to \infty$ is the final control time.

In order to identify the parameters of (41), (42), the beam is fixed at the point of $L = 0.20$ m from the joint. Using the rectangular reference angular velocity $\dot{\theta}_d$ with the amplitude of 2.0×10^{-4} rad/s and the period of 0.5 s, the joint torque is measured (sampling frequency: 100 Hz). The measured results are shown in Fig. 5 as the fine line. On the other hand, the response of the identified model with the parameters a_1=69.3 and b_0=8.24 is shown as the thick line. From the Fig. 5, we can see that the error between

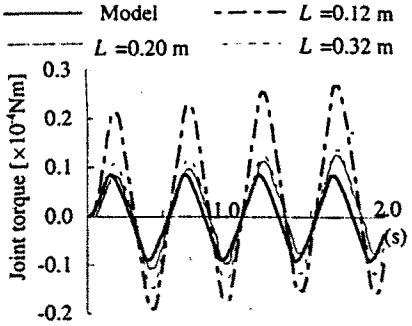

Figure 5: Responses of the flexible beam and the identified parameters

the responses of the identified model and the flexible beam increases with control time.

With the same experimental device, the fixed position L of the beam is changed. The measured result is too shown in the Fig. 5: the dashed line represents the result with $L = 0.32$ m, and the dotted line represents the result with $L = 0.12$ m. When the position L is changed, the joint torque is largely changed and quite different from the response of the identified model with the parameters of $L = 0.20$ m.

3.3 Experimental Result

The Linear Quadratic Regulator (LQR) is designed using the weight matrices

$$Q = \begin{bmatrix} 4 & 0 \\ 0 & 0 \end{bmatrix}, \quad R = 0.5,$$

(See (43)). The solution P_n of the Riccati equation, the transformation matrix Θ, the feedback gain K_n and the compensatory gain Δ_K are respectively given as follows:

$$P_n = \begin{bmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \end{bmatrix} = \begin{bmatrix} 98.144 & 1.414 \\ 1.414 & 0.049 \end{bmatrix},$$

$$\Theta = \begin{bmatrix} \theta_{11} & \theta_{12} \\ \theta_{21} & \theta_{22} \end{bmatrix} = \begin{bmatrix} -0.02 & -7.49 \\ 69.41 & 3.19 \end{bmatrix},$$

$$K_n = R^{-1} B^T P_n = 2[p_{21} \quad p_{22}], \qquad (44)$$

$$\Delta_K = R^{-1} B^T \Theta = 2[\theta_{21} \quad \theta_{22}]. \qquad (45)$$

From (16) the optimal input $u^*(t)$ can be calculated as

$$\begin{aligned} u^*(t) \;=\; & -2[(p_{21} x_1(t) + p_{22} x_2(t)) \\ & + (\theta_{21} x_{NN_1} + \theta_{22} x_{NN_2})] \end{aligned} \qquad (46)$$

where x_{NN_i} represents the ith output of the neural network.

In the experiments, two units in the input layer, eight units in the hidden layer and two units in the output layer are used. The initial value of the weight is chosen as an uniform random number in $[-1.0 \times$

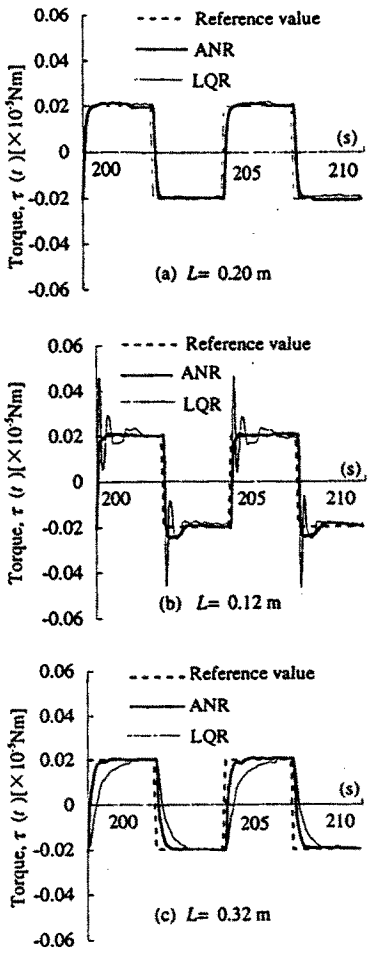

Figure 6: Experimental results of the torque control

$10^{-3}, +1.0 \times 10^{-3}]$, the learning rate is η=0.5, and the parameter γ of the sigmoid function is γ=1.

In the control system, the reference value is a rectangular signal where the amplitude is 2.0×10^{-5} Nm, the period is 5 s, and the control time is 210 s. The proposed regulator (ANR) is applied to six kinds of the contact location of $L = 0.12$ m, 0.16 m, 0.20 m, 0.24 m, 0.28 m, 0.32 m. Note that the same linear model (41), (42) is used with the parameters identified for $L = 0.20$ m.

Figure 6 shows the experimental results that correspond to $L = 0.20$ m, 0.12 m, 0.32 m, where the fine lines represent the results under the LQR, the thick lines represent the results under the ANR. In Fig. 6(a), since the same contact location L is used for identifying the linear model, the experimental results obtained with the use of the LQR and the ANR are not largely different. However, when L is changed (Fig. 6(b), (c)), the LQR results significant overshoot or undershoot. The ANR with the linear model of $L = 0.20$ m always produces stable responses.

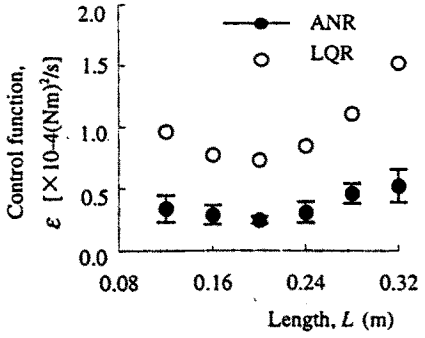

Figure 7: Change of the control performance with the beam length

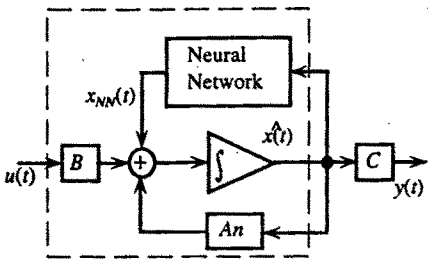

Figure 8: Identification system of the flexible beam

Figure 7 shows the mean squared error $\mathcal{E} = \frac{1}{10}\int_{200}^{210} x_1^2(t)dt$ between 200 s and 210 s. In Fig. 7, the black and white circles represent the results of the ANR and the LQR, respectively. Note that the errors corresponding to the ANR are shown by their mean values and the standard deviations. These errors are calculated for 10 different initial values of the weights. From the result of the ANR, it can be concluded that even if there is large error between the identified parameters and real parameters of the flexible beam, the stable response is always obtained.

When the identified error becomes zero by the training of the neural network, the state $\hat{x}(t)$ of the identification system shown in Fig. 8 should agree with the state $x(t)$ of the torque control system. Fig. 9 shows the measured and identified torque for the input with the amplitude of 2.0×10^{-5} Nm and the period of 5 s ($L = 0.32$ m). The measured result is shown as the fine line and the output of the identification system is shown as the thick line. Comparing the fine and thick lines, we can see that the two lines are almost the same.

4 Conclusions

In this paper, the adaptive neural regulator is proposed. In the proposed regulator scheme, the neural network can identify the unknown part of the controlled system and compensate the control input from the LQR simultaneously. By the training process of the neural network, the control input can be adaptively modified and the identification system is automatically organized. If the learning rate is suitably

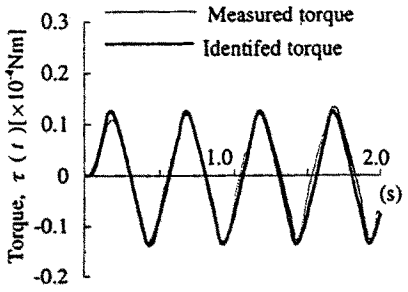

Figure 9: Responses of the Identification system and the flexible beam

chosen, the stability of the proposed regulator scheme can be guaranteed. Also, the experimental results of the torque control for the flexible beam illustrate the effectiveness and applicability of the proposed regulator.

In order to improve the performance of the control system using the neural network, this paper has concentrated on the control structure of the system. Future research will be diverted to the other way to improve the control performance, that is revision of the neural network model using a radial basis function network [13] and so on.

Appendix A:

First, from (13) we have

$$\Delta_P \mathcal{D} + \mathcal{F}\Delta_P = \Delta_A^T P_n + P_n \Delta_A \quad (47)$$

$$\mathcal{D} = BR^{-1}B^T P_n - A_n \quad (48)$$

$$\mathcal{F} = P_n B R^{-1} B^T - A_n \quad (49)$$

Expanding the both side of (47) leads to the linear equation of the following form

$$\Pi_1 \, \mathrm{cs}\, \Delta_P = \Pi_2 \, \mathrm{cs}\, \Delta_A, \quad (50)$$

where

$$\Pi_1 = \mathcal{D}^T \bigotimes I_n + I_n \bigotimes \mathcal{F},$$

$$\Pi_2 = I_n \bigotimes P_n^T + I_n \bigotimes P_n,$$

$\bigotimes$ represents the Kronecker product, $I_n \in \Re^{n\times n}$ is the unit matrix.

When Π_1 is assumed to be a non-singular matrix, $\mathrm{cs}\,\Delta_P$ can be described as

$$\mathrm{cs}\,\Delta_P = \Pi \, \mathrm{cs}\, \Delta_A, \quad (51)$$

$$\Pi = \Pi_1^{-1}\Pi_2. \quad (52)$$

Here, the elements $\Delta_{a_{ij}}$ of the uncertainties Δ_A for $j(=1,2,\cdots,n)$ are assumed to be equal, we have

$$\Delta_P = \Theta \Delta_A, \quad (53)$$

where $\Theta \in \Re^{n\times n}$ is the transformation matrix.

Appendix B:

From the error back-propagation algorithm, the weight updating rules (32), (33) can be rewritten by the matrix form

$$V(t+\Delta t) = V(t) - \eta \frac{\partial E(t)}{\partial V(t)}, \quad (54)$$

$$W(t+\Delta t) = W(t) - \eta \frac{\partial E(t)}{\partial W(t)}, \quad (55)$$

where $[\partial E(t)/\partial V(t)] \in \Re^{n\times p}$, $[\partial E(t)/\partial W(t)] \in \Re^{p\times n}$ are the matrices, elements of which are the second terms of (32), (33), respectively. Also we can see

$$\frac{\partial E(t)}{\partial V(t)} = \frac{\partial E(t)}{\partial \hat{x}(t)} \left[\frac{\partial \hat{x}(t)}{\partial x_{NN}(t)}\right]^T \frac{\partial x_{NN}(t)}{\partial V(t)}. \quad (56)$$

By (19) we have

$$\frac{\partial E(t)}{\partial \hat{x}(t)} = \frac{\partial}{\partial \hat{x}(t)}\{\frac{1}{2}[\hat{x}(t)-x(t)]^T[\hat{x}(t)-x(t)]\} = \epsilon(t) \in \Re^{n\times 1}. \quad (57)$$

Moreover, from (15) and (21) we can easily lead

$$\hat{x}(t) = \int_0^t [A_n - BK_n]x(\tau)d\tau + \Lambda(t), \quad (58)$$

$$\Lambda(t) = \int_0^t [I_n - B\Delta_K]x_{NN}(\tau)d\tau = [\lambda_1,\cdots,\lambda_n]^T \in \Re^{n\times 1}, \quad (59)$$

so that $\partial\hat{x}(t)/\partial x_{NN}(t)$ and $\partial x_{NN}(t)/\partial V(t)$ can be represented as [12]

$$\frac{\partial \hat{x}(t)}{\partial x_{NN}(t)} = \frac{\partial \Lambda(t)}{\partial x_{NN}(t)} \in \Re^{n^2\times 1}, \quad (60)$$

$$\frac{\partial x_{NN}(t)}{\partial V(t)} = [\frac{\partial x_{NN_1}(t)}{\partial V(t)},\cdots,\frac{\partial x_{NN_n}(t)}{\partial V(t)}]^T \in \Re^{n^2\times p}. \quad (61)$$

By (34), near the optimal set of the weights, $\partial x_{NN}(t)/\partial V(t)$ reduces to

$$\frac{\partial x_{NN}(t)}{\partial V(t)} = \varrho H x^T(t) W^T(t), \quad (62)$$

where $H = [10\cdots0\vdots01\cdots0\vdots\cdots\vdots00\cdots1]^T \in \Re^{n^2\times 1}$ is the vector that the $[(i-1)n+i]$th elements are 1 and all other elements are 0.

Consequently, (56) becomes

$$\frac{\partial E(t)}{\partial V(t)} = \Upsilon_V(t), \quad (63)$$

$$\Upsilon_V(t) = \varrho\epsilon(t)\Gamma^T(t)Hx^T(t)W^T(t) \in \Re^{n\times p}, \quad (64)$$

$$\Gamma(t) = \frac{\partial \hat{x}(t)}{\partial x_{NN}(t)} \in \Re^{n^2\times 1}, \quad (65)$$

and the weight updating rule (54) can be written as

$$V(t+\Delta t) \approx V(t) - \eta \Upsilon_V(t). \quad (66)$$

Also, the weight updating rule (55) can be approximately presented by

$$W(t+\Delta t) \approx W(t) - \eta \Upsilon_W(t), \quad (67)$$

$$\Upsilon_W(t) = \varrho[V^T(t)(x^T(t) \bigotimes I_n)\Gamma(t)\epsilon^T(t)] \in \Re^{p \times n}.$$

Then, from (66) and (67) we can obtain the following form

$$\begin{aligned} & V^T(t+\Delta t)W(t+\Delta t) \\ & \approx V^T(t)W(t) - \eta Q_1(t) + \eta^2 Q_2(t), \quad (68) \\ Q_1(t) &= \Upsilon_V^T(t)W(t) + V^T(t)\Upsilon_W(t), \quad (69) \\ Q_2(t) &= \Upsilon_V^T(t)\Upsilon_W(t). \quad (70) \end{aligned}$$

From (36), (68), $\Phi(t+\Delta t)$ can be described as

$$\begin{aligned} \Phi(t+\Delta t) &= \varrho V^T(t+\Delta t)W(t+\Delta t) - \Delta_A \\ &\approx \Phi(t) - \eta\varrho Q_3(t), \quad (71) \\ Q_3(t) &= Q_1(t) - \eta Q_2(t), \quad (72) \end{aligned}$$

and by (37) $\Psi(t+\Delta t)$ can be given as

$$\begin{aligned} \Psi(t+\Delta t) &= [\mathrm{cs}\Phi(t)]^T \mathrm{cs}\Phi(t) + (\eta\varrho)^2\rho_6 \\ & \quad -2\eta\varrho[\mathrm{cs}Q_3(t)]^T \mathrm{cs}\Phi(t), \quad (73) \end{aligned}$$

where $\rho_6 = [\mathrm{cs}Q_3(t)]^T \mathrm{cs}Q_3(t) = \|Q_3(t)\|_2 \cdot \|Q_3(t)\|_2$ represents the matrix norm of $Q_3(t)$.

From (38), (73), the difference Δ_Ψ can be derived as

$$\Delta_\Psi = -\eta\varrho\{2[\mathrm{cs}Q_3(t)]^T \mathrm{cs}\Phi(t) + \eta\varrho\rho_6\}. \quad (74)$$

When the difference $\Delta_\Psi < 0$, the asymptotic stability of the parameter error $\Phi(t)$ can be guaranteed. So, by (72), (74) we can derive the following condition:

$$\rho_5\{[\mathrm{cs}Q_1(t)]^T - \eta[\mathrm{cs}Q_2(t)]^T\}\mathrm{cs}\Phi(t) > \eta > 0, \quad (75)$$

$$\rho_5 = 2/(\varrho\rho_6).$$

Setting

$$\rho_1 = \rho_5\rho_3, \quad \rho_2 = \rho_5\rho_4,$$

$$\rho_3 = [\mathrm{cs}Q_1(t)]^T \mathrm{cs}\Phi(t),$$

$$\rho_4 = [\mathrm{cs}Q_2(t)]^T \mathrm{cs}\Phi(t),$$

and assuming $\rho_3 > 0$ and $\rho_4 > -1/\rho_5$, or $\rho_3 < 0$ and $\rho_4 < -1/\rho_5$, we can obtain the sufficient condition:

$$\rho_1/(1+\rho_2) > \eta > 0. \quad (76)$$

References

[1] J. Douglas and M. Athans: Robust Linear Quadratic Design with Real Parameter Uncertainty, *IEEE Trans. On Automatic Control*, AC Vol.39-1, pp107-111, 1994

[2] T.Yamada and T.Yabuta: Nonlinear Neural Network Controller for Dynamic Systems, *Proceedings of 16th Annual Conference of IEEE Industrial Electronics Society (IECON'90)*, pp1244-1249, 1990

[3] T.Yamada and T.Yabuta: Some Remarks on Characteristics of Direct Neuro-Controller with regard to Adaptive Control, *Trans. On the Society of Instrument and Control Engineers*, Vol.27, No.7, pp784-791, 1991

[4] K. Takahashi: Neural-Network-Based Learning Control Applied to a Single-Link Flexible Arm, *Proc. of Second International Conference on Motion and Vibration Control*, pp811-816, 1994

[5] M. M. Polycarpou and A. J. Helmicki: Automated Fault Detection and Accommodation:A learning Systems Approach, *IEEE Trans. On Systems, Man, and Cybernetics*, SMC Vol.25-11, pp1447-1458, 1995

[6] G.A. Rovithakis and M. A. Christodoulou: Direct Adaptive Regulation of Unknown Nonlinear dynamical Systems via Dynamic Neural Networks, *IEEE Trans. On Systems, Man, and Cybernetics*, SMC Vol.25-12, pp1578-1594, 1995

[7] T. Kailath: Linear Systems, Prentice-Hall, Inc.,Englewood Cliffs , 1980

[8] D. E. Rumelhart, G. E. Hinton and R. J. Williams: Learning Representations by Error Propagation, In D.E.Rumelhart, J.L.McClelland & PDP Research Group: Parallel Distributed Processing, Vol.1, MIT Press, 1986

[9] G.Cybenko: Approximation by Superposition of a Sigmoidel Function, *Math. Control Signal Systems*, Vol.2, pp303-314, 1989

[10] M. Kaneko, N. Kanayama, and T. Tsuji: 3-D Active Antenna for Contact Sensing, *Proc. of the 1995 IEEE Int. Conference on Robotics and Automation*, May, Vol.1, pp1113-1119, 1995

[11] B. H. Xu, T. Tsuji and M. Kaneko: Adaptive Neural Controller for a Class of Plant with Nonlinear Uncertainties, *The IEEE 4the Int. Workshop on Advanced Motion Control*,pp293-298, 1996

[12] J.R.Magnus and H.Neudecker: Matrix Differential Calculus with Applications in Statistics and Econometrics, New York:Wiley, 1988

[13] D. Gorinevsky, A. Kapitanouvsky and A. Goldenberg: RBF Network Architecture for Motion Planning and Attitude Stabilization of Nonholonomic Spacecraft/Manipulators Systems, *Proceedings of IEEE International Conference on Robotics and Automation*, pp2749-2754, 1994

IEEE Catalog Number:	96CH35908	
ISBN:	0-7803-3213-X	(Softbound Edition)
	0-7803-3214-8	(Casebound Edition)
	0-7803-3215-6	(Microfiche Edition)
Library of Congress Number:	96-75376	