Adaptive Neural Network Control for Robot Manipulators

Min-Jung Lee and Young-Kiu Choi

Abstract - In the recent years neural networks have fulfilled the promise of providing model-free learning controllers for nonlinear systems; however, it is very difficult to guarantee the stability and robustness of neural network control systems. This paper proposes an adaptive neural network control for robot manipulators based on the radial basis function network (RBFN). The RBFN is a branch of the neural networks and is mathematically tractable. So we adopt the RBFN to approximate nonlinear robot dynamics. The RBFN generates control input signals based on the Lyapunov stability that is often used in the conventional control schemes. The saturation function is also chosen as an auxiliary controller to guarantee the stability and robustness of the control system under the external disturbances and modeling uncertainties.
Keywords - Radial Basis Function Network(RBFN), Lyapunov stability, robot manipulator, stability and robustness

1. Introduction

Robot manipulators commonly used as industrial automatic elements are systems with high nonlinearities often unknown and time-varying. To design a controller for robot manipulators, it is necessary to have the exact trajectory tracking performance for reference inputs and the robustness for the external disturbances. The conventional feedback controllers such as PID controllers, are commonly used in the field of industries because their control architectures are very simple and easy to implement. However, when these conventional feedback controllers are directly applied to nonlinear systems, they suffer from the poor performance and low robustness due to the unknown nonlinearities and the external disturbances. To deal with the unknown nonlinearities and external disturbances, various control strategies are proposed in the forms of the automatic tuning of PID control, variable structure control, feedback linearization, adaptive control, intelligent control, etc [1-5].

For some decades, adaptive control schemes started in the early 1950s are applied to the various nonlinear systems since these control schemes can be automatically adjusting the controller parameters in the face of changing system dynamics. On-line adaptation laws are used to estimate the unknown parameters of the system, and appropriate controllers are designed to control the system in order to satisfy the desired performance. In spite of the impressive applications made to several practical problems and the sophisticated mathematical machinery underlying the development of adaptive control algorithms, an essential characteristic of these procedures is their model dependence, i.e., the requirement for explicit a priori specified model structures [6].

Intelligent control approaches such as neural networks and fuzzy inference systems do not require mathematical models and have an ability to approximate nonlinear systems. With these features of intelligent control theory, many researchers have been attempting to use intelligent control approaches to represent complex plants and construct advanced controllers such as the model reference or direct adaptive controllers [7,8]. Especially, in the recent years researches on neural networks have been pursued to provide model-free learning controllers for a class of nonlinear systems [9-17]. Narendra and Parthasarathy [9] proposed the dynamic back-propagation algorithm for identification and control employing the multilayer perceptron (MLP). Chen and Khail [11] considered the MLP in the adaptive control of feedback, linearizable minimum phase plants represented by an input-output model. Karakasoglu et al. [6] used the MLP for the on-line adaptive control of complex dynamical systems characterized by an interconnection of several subsystems. Revithakis and Christodoulou [12] considered the dynamical identifier to perform identification with the MLP and constructed dynamic state feedback controller. Lewis et al.[13] proposed a neural net robot controller. Choi et al. [14] presented a neural network compensator for conventional control systems to improve the control performance without hardware modifications. Meantime the radial basis function network (RBFN) is also applied to control dynamic systems [15-17]. The structure of the RBFN is simpler than that of the MLP, so the learning speed of the RBFN is generally faster than that of the MLP. Also the RBFN is mathematically tractable.

Zhihong et al. [15] used the RBFN for a robust adaptive tracking control scheme for a class of nonlinear systems. Instead of learning directly the system uncertainty, the RBFN is used to adjust adaptively the gain of the sliding

Manuscript received; Sep. 12, 2001 accepted: Jan. 18, 2002.
Min-Jung Lee is with Dept. of Electrical Eng., Pusan National University, Pusan, Korea.
Young-Kiu Choi is with the Research Institute of Computer, Information and Communication. He is also with School of Electrical and Computer Eng., Pusan National University, Pusan, Korea.

mode. Patino and Liu [16] developed a neural network controller based on model reference adaptive control. The nonlinear part of the controller, which compensates for the plant nonlinearity, can be implemented by either the RBFN or a feedfoward neural network. And Seshagiri and Khail [17] presented an adaptive output feedback control scheme for the output tracking of a class of continuous-time nonlinear plants. The RBFN is used to compensate adaptively for the plant nonlinearities.

In the above researches related to the RBFN, plants are assumed to be single-input-single-output systems, both centers and widths of Gaussian functions in the RBFN are given and fixed, and their theoretical results are verified through simulation studies for simple systems.

This paper proposes another adaptive neural network controller based on the RBFN. It deals with tracking control problems for robot manipulators that are multi-input-multi-output systems, and the adaptation laws for Gaussian functions are also derived to adjust the centers and widths of the RBFN in real-time. The robot dynamics expressed in terms of the filtered tracking errors is the nonlinear function. The RBFN is applied to approximate the nonlinear function. The adaptation laws are derived to guarantee the stability of the control system based on the Lyapunov method that is often used in the conventional adaptive control scheme. Also, to guarantee the stability and robustness of control system under the existence of nonlinearities and external disturbances, a saturation function is employed as an auxiliary controller.

Finally, the proposed adaptive neural network control scheme is applied to the SCARA-type robot manipulator. We can find experimentally the validity of the adaptive neural network control scheme by comparing with other control strategies.

2. Dynamics and Structural Properties of Robot Manipulators

The dynamics of an n-link robotic manipulator may be expressed in the Euler-Lagrangian form [1,2,13].

$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) + \tau_d = \tau \tag{1}$$

where $q \in R^n$ is the joint variable vector, $D(q) \in R^{n\times n}$ is the inertial matrix, $C(q,\dot{q})\dot{q} \in R^n$ is the Coriolis/centrifugal force, and $G(q) \in R^n$ is the gravity vector. $\tau \in R^n$ denotes the vector of applied joint torques and $\tau_d \in R^n$ is the disturbance vector which could represent modeling errors.

In general, robot dynamics have the following properties. The properties hold for all rigid-link manipulators [1,2,13].

Property 1: The inertial matrix $D(q)$ is symmetric and positive definite, and there exist scalars d_1 and d_2 such that

$$d_1 I \le D(q) \le d_2 I .$$

Property 2: The Coriolis/centrifugal force $C(q,\dot{q})\dot{q}$ is bounded by $c_b(q)\|\dot{q}\|^2$ with $c_b(q) \in C^1(S)$. S is a simply connected compact set of R^n.

Property 3: The matrix $\dot{D} - 2C$ is skew-symmetric, that is, the matrix is satisfied with $x^T(\dot{D} - 2C)x = 0 \quad \forall x \in R^n$. Strongly related to the skew symmetry is the passivity.

Property 4: The unknown disturbance satisfies $\|\tau_d\| < b_d$ with b_d a positive constant bound.

3. Radial Basis Function Networks

The locally tuned and overlapped receptive field is a well-known structure that has been studied in regions of the cerebral cortex, visual cortex, and so on. Based on the biological receptive fields, Moody and Darken proposed a network structure, namely, a RBNF that employs local receptive fields to perform function mapping. The RBNF is well known in the field of approximation of nonlinear function and pattern recognition. Especially the RBFN has a faster convergence property than the multilayer perceptron because the RBFN has a simple architecture. The RBFN also has a similar feature to the fuzzy inference system. First, the output value is calculated using the weighted sum or weighted average method. Second, the number of hidden layer's nodes of the RBFN is the same as the number of if-then rules in the fuzzy inference system. Third, the receptive field functions of the RBFN are similar to the membership functions of fuzzy inference system's premise part. Fig. 1 shows the architecture of the RBFN with M receptive field units, where the j-th receptive field unit is usually a Gaussian function or a logistic function [17, 18].

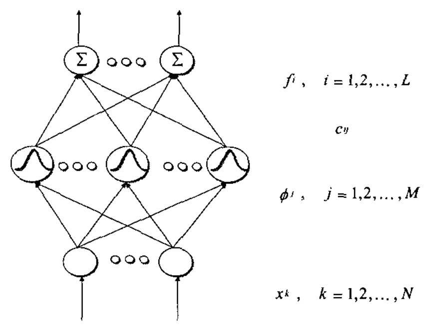


Fig. 1 Architecture of the RBFN with M receptive field units.

If we select Gaussian function as the receptive field unit and use the weighted sum method to calculate the output of the RBFN, then its output is

$$f_i = \sum_{j=1}^{M} c_{ij}\phi_j \qquad i = 1,2,\cdots,L \tag{2}$$

$$\phi_j(x) = \exp\left(-\frac{\|x - m_j\|^2}{\sigma_j^2}\right) \tag{3}$$

where M and L are the numbers of hidden nodes and output nodes, respectively. Note that c_{ij} is the weight connecting the j-th hidden node to the i-th output node, $\phi_j(x)$ is the j-th Gaussian function, $m_j \in R^n$ is the center vector, and $\sigma_j \in R^n$ is the j-th standard deviation.

In this paper, the RBFN is used to design an adaptive controller because the structure is simpler than the multilayer perceptron. As the design of an adaptive control, the RBNF is used as a nonlinear approximator. If the nonlinear function in the forms of (1) is approximated using the RBFN, the approximated system model can be described as

$$f = c^T \Phi + \varepsilon \tag{4}$$

where $f = [f_1 \;\; f_2 \;\; \cdots \;\; f_L]^T$, $\Phi = [\phi_1 \;\; \phi_2 \;\; \cdots \;\; \phi_M]^T$,

$$c^T = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1M} \\ c_{21} & c_{22} & \cdots & c_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ c_{L1} & c_{L2} & \cdots & c_{LM} \end{bmatrix}, \text{ and } \varepsilon = [\varepsilon_1 \;\; \varepsilon_2 \;\; \cdots \;\; \varepsilon_L]^T.$$

In (4), an approximation error vector ε exists because we just consider the finite dimensional hidden nodes of the RBFN. But approximation error can be made a very small value. We can see that the norm value of ε is bounded by a constant value such that

$$\|\varepsilon\| \le \varepsilon_N. \tag{5}$$

where $\|\cdot\|_1$ denotes the 1-norm.

4. Adaptive Neural Network Controller

4.1 Architecture

This paper attempts to connect neural networks to adaptive control schemes in order to solve the difficult problems such as the stability in neural network control systems and the requirement of the model structure in the adaptive control scheme. We choose the RBFN since its architecture is simple and mathematically tractable. Fig. 2 shows the block diagram of the adaptive neural network controller. The proposed adaptive neural network controller consists of two parts: a nonlinear function approximator and an auxiliary controller. In the nonlinear function approximator, the RBFN represents the nonlinear robot dynamics written in terms of the filtered tracking errors. The adaptation laws for updating the weights of the RBFN are derived to guarantee the stability of control system. This approach is somewhat different from the conventional adaptive control schemes because the conventional adaptive control schemes commonly use the linear reparameterization in the unknown terms of robot manipulators. Next, we have the auxiliary controller to guarantee the stability and robustness under the existence of nonlinearities and external disturbances.

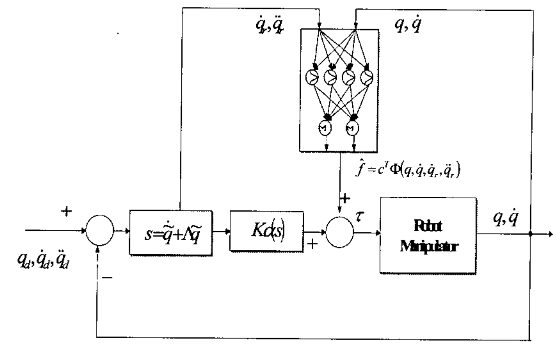


Fig. 2 Block diagram of the adaptive neural network controller.

4.2 Stability and Robustness

To derive the adaptation law of the RBFN, the nonlinear function $\alpha(s)$ in Fig. 2 is selected to be the signum function. And then the form of $\alpha(s)$ is extended to cover the saturation function. Now consider the robot dynamics in the form of (1).

If the reference trajectory, $q_d \in R^n$, is given, the tracking error $\tilde{q}$ is defined by

$$\tilde{q} = q_d - q. \tag{6}$$

The filtered tracking error s and the control input τ are also defined as below.

$$s = \dot{\tilde{q}} + \Lambda \tilde{q} = \dot{q}_r - \dot{q} \tag{7}$$

$$\tau = \hat{c}^T \Phi(x) + K \,\mathrm{sgn}(s) \tag{8}$$

where $x = [q, \dot{q}, \dot{q}_r, \ddot{q}_r]$, $\dot{q}_r = \dot{q}_d + \Lambda \tilde{q}$, $\Lambda = \Lambda^T > 0$, and K is diagonal and positive definition.

Fig. 2 depicts (8). According to the Lyapunov stability analysis, the system is stable if the Lyapunov function is positive-definite and its derivative is negative semi-definite. Therefore, to guarantee the stability of the total control system, a positive-definite Lyapunov function candidate is selected as follows:

$$V = \tfrac{1}{2}\left\{ s^T D s + tr\left(\tilde{c}^T \Gamma_1^{-1} \tilde{c}\right) + tr\left(\tilde{m}^T \Gamma_2^{-1} \tilde{m}\right) + tr\left(\tilde{\sigma}^T \Gamma_3^{-1} \tilde{\sigma}\right) \right\} \tag{9}$$

where $\tilde{c} = c^* - \tilde{c}$ is weight matrix error between the optimal weight matrix c^* and the estimated weight matrix $\hat{c}$ of the RBFN in (8). Also $\tilde{m} = m^* - \hat{m}$ and $\tilde{\sigma} = \sigma^* - \hat{\sigma}$ are the center vector error and standard deviation vector error in (3), respectively. Γ_1, Γ_2, and Γ_3 are diagonal, symmetric and positive-definite matrices.

Differentiating (9) with respect to time, we get

$$\dot{V} = s^T D \dot{s} + \tfrac{1}{2} s^T \dot{D} s - tr\left(\tilde{c}^T \Gamma_1^{-1} \dot{\hat{c}}\right) - tr\left(\tilde{m}^T \Gamma_2^{-1} \dot{\hat{m}}\right) - tr\left(\tilde{\sigma}^T \Gamma_3^{-1} \dot{\hat{\sigma}}\right). \tag{10}$$

If we differentiates s with respect to time, the robot dynamics (1) may be written in terms of the filtered tracking errors as follows:

$$D\dot{s} = D\ddot{q}_r + C\dot{q}_r + G + \tau_d - Cs - \tau. \tag{11}$$

Substituting (11) into (10), we have

$$\dot{V} = s^T\left[D\ddot{q}_r + C\dot{q}_r + G + \tau_d - \tau\right] + \tfrac{1}{2}s^T\left(\dot{D} - 2C\right)s - tr\left(\tilde{c}^T\Gamma_1^{-1}\dot{\hat{c}}\right)$$
$$- tr\left(\tilde{m}^T\Gamma_2^{-1}\dot{\hat{m}}\right) - tr\left(\tilde{\sigma}^T\Gamma_3^{-1}\dot{\hat{\sigma}}\right). \tag{12}$$

Using the property 3: $s^T\left(\dot{D} - 2c\right)s = 0 \quad \forall s \in R^n$, (12) becomes

$$\dot{V} = s^T\left[\{D\ddot{q}_r + C\dot{q}_r + G\} + \tau_d - \tau\right] - tr\left(\tilde{c}^T\Gamma_1^{-1}\dot{\hat{c}}\right) - tr\left(\tilde{m}^T\Gamma_2^{-1}\dot{\hat{m}}\right)$$
$$- tr\left(\tilde{\sigma}^T\Gamma_3^{-1}\dot{\hat{\sigma}}\right). \tag{13}$$

To approximate the nonlinear robot dynamics using the RBFN, let us define a function as below:

$$f(q, \dot{q}, \dot{q}_r, \ddot{q}_r) = D(q)\ddot{q}_r + C(q, \dot{q})\dot{q}_r + G(q). \tag{14}$$

Approximating the above function using the RBFN with finite hidden nodes shown in (4), we have

$$f(q, \dot{q}, \dot{q}_r, \ddot{q}_r) \approx c^{*T}\Phi(q, \dot{q}, \dot{q}_r, \ddot{q}_r) + \varepsilon. \tag{15}$$

Substituting (8) and (15) into (13) yields

$$\dot{V} = s^T\{\varepsilon + \tau_d - K\,\mathrm{sgn}(s)\} + tr\left\{\tilde{c}^T\left(\Phi s^T - \Gamma_1^{-1}\dot{\hat{c}}\right)\right\} - tr\left(\tilde{m}^T\Gamma_2^{-1}\dot{\hat{m}}\right)$$
$$- tr\left(\tilde{\sigma}^T\Gamma_3^{-1}\dot{\hat{\sigma}}\right). \tag{16}$$

Since it is desirable to have $\dot{V}$ at least negative semi-definite, let us have the following adaptation laws:

$$\dot{\hat{c}} = \Gamma_1\Phi(q, \dot{q}, \dot{q}_r, \ddot{q}_r)s^T \tag{17}$$

$$\dot{\hat{m}} = -\Gamma_2\|s\|_1\hat{m} \tag{18}$$

$$\dot{\hat{\sigma}} = -\Gamma_3\|s\|_1\hat{\sigma} \tag{19}$$

Then (16) becomes

$$\dot{V} \le \sum_{i=1}^{n}\left[|s_i|\left\{\|\varepsilon + \tau_d\|_1 - K_i + tr\left\{\tilde{m}^T\left(m^* - \tilde{m}\right)\right\}\right.\right.$$
$$\left.\left. + tr\left\{\tilde{\sigma}^T\left(\sigma^* - \tilde{\sigma}\right)\right\}\right\}\right]. \tag{20}$$

where K_i is the *i-ith* element of the diagonal matrix K.

From the property of Frobenius norm,

$$tr\left\{\tilde{m}^T\left(m^* - \tilde{m}\right)\right\} = \left\langle\tilde{m}, m^*\right\rangle_F - \|\tilde{m}\|_F^2 \le \|\tilde{m}\|_F\|m^*\|_F - \|\tilde{m}\|_F^2 \tag{21}$$

$$tr\left\{\tilde{\sigma}^T\left(\sigma^* - \tilde{\sigma}\right)\right\} = \left\langle\tilde{\sigma}, \sigma^*\right\rangle_F - \|\tilde{\sigma}\|_F^2 \le \|\tilde{\sigma}\|_F\|\sigma^*\|_F - \|\tilde{\sigma}\|_F^2. \tag{22}$$

Substituting (21) and (22) into (20) we get

$$\dot{V} \le -\sum_{i=1}^{n}\left[|s_i|\left\{K_i - \|\varepsilon + \tau_d\|_1 + \left(\|\tilde{m}\|_F - \frac{m_{\max}}{2}\right)^2 - \frac{m_{\max}^2}{4}\right.\right.$$
$$\left.\left. + \left(\|\tilde{\sigma}\|_F - \frac{\sigma_{\max}}{2}\right)^2 - \frac{\sigma_{\max}^2}{4}\right\}\right] \tag{23}$$

where $m_{\max}$ and $\sigma_{\max}$ are the maximum values of the Frobenius norm of center and standard deviation vector in the RBFN.

The approximation error term ε is limited by the upper bound ε_N as shown in (5) and the disturbance term τ_d by an upper bound b_d. If gain K_i is selected to satisfy the following inequality:

$$K_i \ge \frac{m_{\max}^2}{4} + \frac{\sigma_{\max}^2}{4} + \left(\varepsilon_N + b_d\right) \quad \forall\, i = i, 2, \cdots, m \tag{24}$$

then we have

$$\dot{V} \le 0. \tag{25}$$

From (9) and (25), the control system is stable based on the Lypunov stability.

4.3 Extension to the Saturation Function

In the previous section, the stability and robustness are guaranteed for the control system with (8). Form (8), the control input has the signum function. But the signum function causes a chattering problem. This chattering may give robot manipulators critical damages since it involves extremely high control activity, and further may excite high-frequency dynamics neglected in the course of modeling. In order to reduce the chattering, we extend the mathematical framework of the previous section to the saturation function. A deadzone compensation is also included because the deadzone effect exists in real nonlinear systems and it makes the system performance deteriorated. Fig. 3 shows the saturation function with a deadzone compensation. The saturation function may be represented as a multiplication of the signum function and $\rho(|s_i|)$, i.e., $sat(x_i) = \mathrm{sgn}(s_i)\cdot\rho(x_i)$.

When the saturation function is used (8) can be rewritten by

$$\tau = \hat{c}^T\Phi(x) + K\,sat(s). \tag{26}$$

To prove the stability and robustness of the control system, the same Lyapunov function candidate as that in (9) is defined and is rewritten as bellow:

$$V = \tfrac{1}{2}\left\{s^TDs + tr\left(\tilde{c}^T\Gamma_1^{-1}\tilde{c}\right) + tr\left(\tilde{m}^T\Gamma_2^{-1}\tilde{m}\right) + tr\left(\tilde{\sigma}^T\Gamma_3^{-1}\tilde{\sigma}\right)\right\}. \tag{27}$$

Differentiating (27) with respect to time, we get

$$\dot{V} = s^T\left(\varepsilon + \tau_d\right) - s^TK\,sat(s) + tr\left\{\tilde{c}^T\left(\Phi s^T - \Gamma_1^{-1}\dot{\hat{c}}\right)\right\}$$
$$- tr\left(\tilde{m}^T\Gamma_2^{-1}\dot{\hat{m}}\right) - tr\left(\tilde{\sigma}^T\Gamma_3^{-1}\dot{\hat{\sigma}}\right). \tag{28}$$

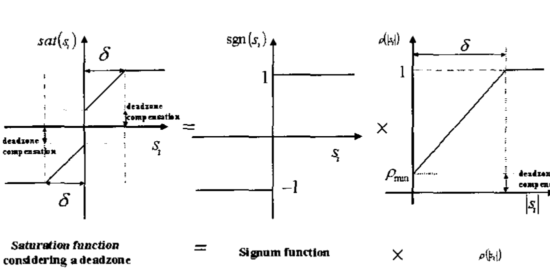


Fig. 3 Saturation function with a deadzone compensation.

Above equation is almost the same as (16). On the other hand, $s^T K\, sat(s)$ can be rewritten in the component base as follows:

$$s^T K\, sat(s) = \sum_{i=1}^{n} K_i \cdot s_i \cdot \text{sgn}(s_i) \cdot \rho(|s_i|) = \sum_{i=1}^{n} K_i \cdot |s_i| \cdot \rho(|s_i|) \tag{29}$$

$$\rho(|s_i|) = \begin{cases} 1 & if \quad |s_i| \ge \delta \\ \dfrac{(1-\rho_{\min})|s_i|}{\delta} + \rho_{\min} & if \quad 0 \le |s_i| < \delta. \end{cases} \tag{30}$$

From (28),(29), and (30) we have

$$\dot V = s^T(\varepsilon + \tau_d) - \sum_{i=1}^{n} K_i |s_i| \rho(|s_i|) + tr\{\tilde c^T(\Phi s^T - \Gamma_1^{-1}\dot{\hat c})\} - tr(\tilde m^T \Gamma_2^{-1} \dot{\hat m}) - tr(\tilde\sigma^T \Gamma_3^{-1} \dot{\hat\sigma}). \tag{31}$$

If the adaptation laws are the same as those defined in (17), (18), and (19), then

$$\dot V \le \sum_{i=1}^{n} \left[|s_i| \left\{ \|\varepsilon + \tau_d\|_1 - K_i \rho(|s_i|) \right\} + tr\{ \tilde m^T (m^* - \tilde m) \} + tr\{ \tilde\sigma^T (\sigma^* - \tilde\sigma) \} \right]. \tag{32}$$

From Fig. 3, we have $\rho_{\min} \le \rho(|s_i|) \le 1$. The lower bound of the function $\rho(|s_i|)$ is a positive constant, $\rho_{\min}$. Therefore we get

$$\dot V \le \sum_{i=1}^{n} \left[|s_i| \left\{ \|\varepsilon + \tau_d\|_1 - K_i \rho_{\min} \right\} + tr\{ \tilde m^T (m^* - \tilde m) \} + tr\{ \tilde\sigma^T (\sigma^* - \tilde\sigma) \} \right]. \tag{33}$$

From the properties of Frobenius norm, (21) and (22),

$$\dot V \le -\sum_{i=1}^{n} \left[|s_i| \left\{ K_i \rho_{\min} - \|\varepsilon + \tau_d\|_1 + \left(\|\tilde m\|_F - \frac{m_{\max}}{2} \right)^2 - \frac{m_{\max}^2}{4} + \left(\|\tilde\sigma\|_F - \frac{\sigma_{\max}}{2} \right)^2 - \frac{\sigma_{\max}^2}{4} \right\} \right]. \tag{34}$$

If the gain is selected to satisfy the following inequality:

$$K_i \ge \frac{\dfrac{m_{\max}^2}{4} + \dfrac{\sigma_{\max}^2}{4} + (\varepsilon_N + b_d)}{\rho_{\min}} \qquad \forall\, i = i, 2, \cdots, m \tag{35}$$

then

$$\dot V \le 0. \tag{36}$$

Finally, if the control gains are selected to satisfy the inequality (35), then the derivative of the Lyapunov function with respect to time becomes negative semi-definite. So, the control system is stable.

5. Experimental Result and Discussions

5.1 Description of the Robot System

The 4-link SCARA-type robot manipulators shown in Fig. 4 is employed as a testbed in this paper. The joints 1 and 2 of the robot manipulator are chosen for experiment. Fig. 5 shows the hardware configuration of robot control system. It consists of an IBM PC, a DSP board to calculate the control inputs online, a DIO board to acquire the error signals and position data, and a D/A board to send command signals to the robot manipulator. The proposed control algorithm is implemented using the C language.

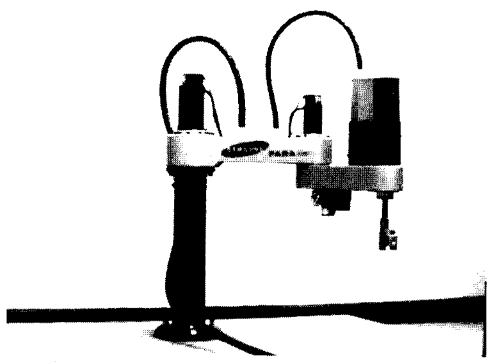


Fig. 4 SCARA-type robot manipulator.

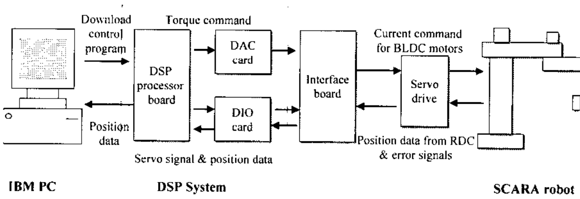


Fig. 5 Hardware configuration of the robot control system.

5.2 Experimental Results

The adaptive neural network controller is compared with two other controllers: the PID controller and the neuro-controller using the MLP [2]. We obtained the PID controller gains by ES such as $K_{P1} = 24.23$, $K_{I1} = 7.27$, $K_{D1} = 0.60$, $K_{P2} = 39.9$, $K_{I2} = 8.34$, and $K_{D2} = 1.78$ for the joints 1 and 2. And in order to experiment the neuro-controller using the MLP, we adopted two and four neurons in the input layer and hidden layer, respectively. The error-backpropagation algorithm is used as an updating algorithm for the neuro-controller using the MLP. The proposed control inputs (26) (17), (18), and (19) are used for the experiment. Control parameters are given in Table I. The numbers of neurons in the input layer and hidden layer are eight and fifteen, respectively. The sampling time is set to be 5[*ms*]. Four different experiments are done to consider the performance under the various environments. The first case is for the sinusoidal reference trajectory with the frequency, $\omega = 1.88$ [*rad/sec*]. The second case is for the reference trajectory two times faster than the first case. The third case is for 4[kg] load with the same frequency as the second case. Finally, the last case is for the circular reference trajectory in the Cartesian space.

Table 1 Control parameters for the adaptive neural network controller.

Parameters	Values
λ	[8, 9]
K	Diag[1.8, 1.9]
Γ_1	0.7×I
Γ_2	0.001×I
Γ_3	0.001×I
δ	Diag[0.5, 0.5]
$\rho_{\min}$	0.3

First, the reference input trajectories are defined as $q_1^r(t) = 0.4\cos(1.88t)[rad]$ and $q_2^r(t) = 0.4\{\sin(1.88t)+1\}[rad]$ for the joints 1 and 2, respectively. Fig. 6 shows the tracking errors of the PID controller, the neuro-controller using the MLP, and the proposed adaptive neural network controller. The neuro-controller using the MLP and the proposed controller can make the tracking errors reduce during learning process since both controllers have the learning ability. But the proposed controller has the faster reduction rate in tracking errors than the neuro-controller using the MLP. Second, another reference input trajectories are defined as $q_1^r(t) = 0.4\cos(3.76t)[rad]$ and $q_2^r(t) = 0.4\{\sin(3.76t)+1\}[rad]$ for the joints 1 and 2, respectively. Fig. 7 shows the tracking errors of the three controllers mentioned above. The performance of the PID controller is deteriorated since the speed of the reference trajectories become two times faster. However, the tracking errors of the neuro-controller using the MLP and the proposed controller become reduced as the time goes on. Third, we have 4[kg] load with the same reference input trajectories as those in the second case. Fig. 8 describes the tracking errors. Due to the learning process, the neuro-controller using the MLP and the proposed controller are less sensitive to the disturbance than the PID controller.

Finally, a circle reference trajectory in the Cartesian space shown in Fig. 9 is applied to the manipulator. Fig. 10 shows the tracking errors of the controllers in the Cartesian space. The tracking errors of the proposed adaptive neural network controller are reduced very fast.

Experimental results mentioned above indicate that the proposed controller is very adaptable to the environmental changes and is more robust than PID controller and the neuro-controller using the MLP.

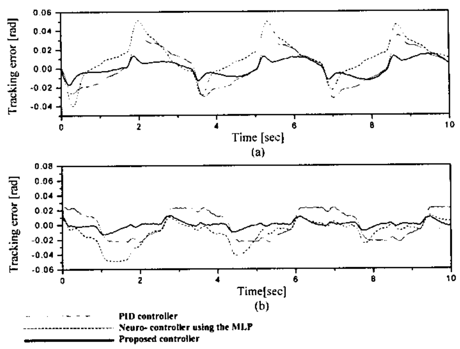


Fig. 6 Tracking errors under reference inputs with $\omega = 1.88[rad/\sec]$. (a) Joint 1. (b) Joint 2.

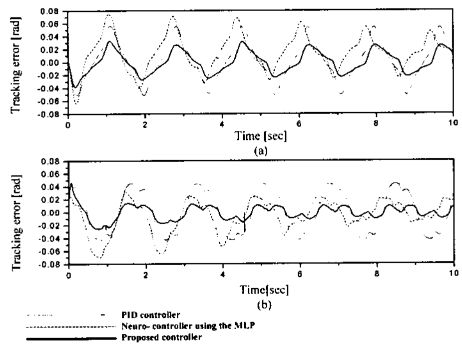


Fig. 7 Tracking errors under reference inputs with $\omega = 3.76[rad/\sec]$. (a) Joint 1. (b) Joint 2.

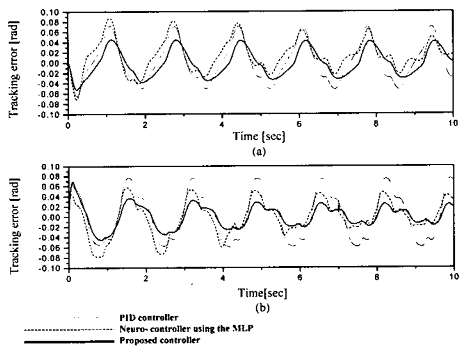


Fig. 8 Tracking errors under disturbance torque. (a) Joint 1. (b) Joint 2.

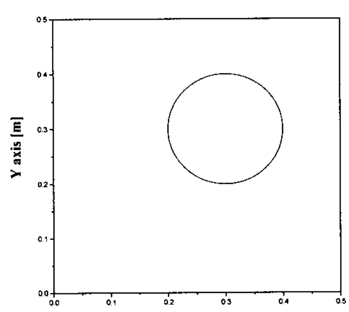


Fig. 9 Circle trajectory in the Cartesian space.

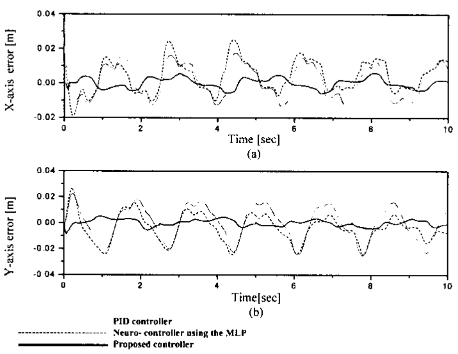


Fig. 10 Tracking errors for the circle trajectory in the Cartesian space. (a) Joint 1. (b) Joint 2.

6. Conclusions

In this paper, we propose an adaptive neural network controller for robot manipulator to solve the problems generally known in the neural network controller, i.e., how to guarantee the stability and robustness of the neural network control system. The structure of the adaptive neural network controller is composed of two parts. The first part is the RBFN approximator. The RBFN generates control input signals based on the Lyapunov method that is often used in the conventional adaptive control method.

To build the first part of the adaptive neural network controller, the robot dynamics are formulated as nonlinear functions in terms of the filtered tracking errors, and the RBFN is employed to approximate the nonlinear functions. This approach is somewhat different from the conventional adaptive control schemes that have the linear reparameterization of robot dynamics. The adaptation laws of the RBFN are derived to guarantee the stability of the control system based on the Lyapunov method. The second part is an auxiliary controller. The auxiliary controller consists of the saturation function in order to guarantee the stability and robustness of the control system under the existence of disturbance and modeling errors.

Finally, the adaptive neural network controller is applied to the SCARA-type robot manipulator to show the validity of the controller by comparing with other control strategies.

References

[1] Jean-Jacques E. Slotine and Weiping Li, *Applied Nonlinear Control*, Prentice-Hall Inc., 1991.

[2] F. L. Lewis, C. T. Abadallah and D. M. Dawson, *Control of Robot Manipulators*, Macmillan Publishing, 1993.

[3] Sigeru Omatu, Marzuki Khalid and Rubiyah Yusof, *Neuro-Control and Its Applications: Advances in Industrial Control*, Springer-Verlag, 1996.

[4] W. J. Daunicht, "Control of manipulators by neural networks," *IEE Proceedings*, Vol. 136, No. 5, pp. 395-399, September 1989.

[5] F. C. Sun, Z. Q. Sun and G. Fėng, "An adaptive fuzzy controller based on sliding mode for robot manipulators," *IEEE Trans. on SMC-Part B: Cybernetics*, Vol. 29, No. 4, pp. 661-667, August 1999.

[6] Ahmet Karakasoglu, Subramania I. Sudharsanan, and Malur K. Sundareshan, "Identification and decentralized adaptive control using dynamical neural networks with application to robotic manipulators," *IEEE Trans. on Neural Networks*, Vol. 4, No. 6, pp. 919-930, November 1993.

[7] Tang-Kai Yin and C. S. George, "Fuzzy model-reference adaptive control," *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. 25, No. 12, pp. 1606-1615, December 1995.

[8] Li-Xin Wang, "Stable adaptive fuzzy controllers with application to inverted pendulum tracking," *IEEE Trans. on Systems, Man, and Cybernetics-Part B: Cybernetics*, Vol. 26, No. 5, pp. 677-681, October 1996.

[9] Kumpati S. Narendra and Kannan Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Trans. on Neural Networks*, Vol. 1, No. 1, pp. 4-27, March 1990.

[10] Robert M. Sanner and Jean-Jacques E. Slotine, "Gaussian networks for direct adaptive control," *IEEE Trans. on Neural Networks*, Vol. 3, No. 6, pp. 837-868, November 1992.

[11] Fu-Chuang Chen and Hassan K. Khail, "Adaptive control of nonlinear systems using neural networks," *Int. J. Control*, Vol. 55, No. 6, pp. 1299-1317, 1992.

[12] George A. Revithakis and Manolis A. Christodoulou, "Adaptive control of unknown plants using dynamical neural networks," *IEEE Trans. on System, Man, and Cybernetics*, Vol. 24, No. 3, pp. 400-411, March 1994.

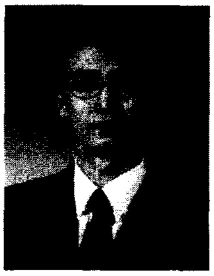
[13] Frank L. Lewis, Kai Liu and Aydin Yeildirek, "Neural net robot controller with guaranteed tracking performance," *IEEE Trans. on Neural Networks*, Vol. 6, No. 3, pp. 703-715, May 1995.

[14] Y.-K. Choi, M.-J. Lee, S. Kim and Y.-C. Kay, "Design and implementation of an adaptive neural-network compensator for control system," *IEEE Trans. On Industrial Electronics*, Vol. 48, NO. 2, pp. 416-423, April 2001.

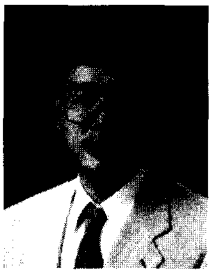
[15] Man Zhihong, H. R. Wu and M. Palaniswami, "An adaptive tracking controller using neural networks for a class of nonlinear systems," *IEEE Trans. on Neural Networks*, Vol. 9, No. 5, pp. 947-955, September 1998.

[16] H. D. Patino and Derong Liu, "Neural network-based model reference adaptive control. system," *IEEE Trans. on Systems, Man, and Cybernetics-Part B:*

Cybernetics, Vol. 30, No. 1, pp. 198-204 February 2000.
[17] Sridhar Seshagiri and Hassan K. Khail, "Output feedback control of nonlinear systems using RBF neural networks," *IEEE Trans. on Neural Networks*, Vol. 11, No. 1, pp. 69-79, January 2000.
[18] Chin-Teng Lin and C. S. George Lee, *Neural Fuzzy Systems*, Prentice -Hall, 1996.
[19] Jack M. Zurada, *Introduction to Artificial Neural Systems*, West, 1992.



Min-Jung Lee received the B.S. degree in electrical engineering from Pukyong National University, Pusan, Korea in 1996, and the M.S. and Ph.D. degrees in electrical engineering from Pusan National University, Pusan in 1998 and 2001. His research interests are intelligent control, evolutionary algorithms, variable structure control and robotics.



Young-Kiu Choi received the B.S. degree in electrical engineering from Seoul National University, Korea in 1980, and the M.S. and Ph.D. degrees in electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST), in 1982 and 1987, respectively. He was a visiting scholar at CALTECH, U.S.A. from 1990 to 1991. He was also a visiting professor at University of Southwestern Louisiana, U.S.A. from 1998 to 1999. He is currently a professor at the school of electrical and computer engineering Pusan National University. His research interests are intelligent control, evolutionary algorithms, variable structure control, robotics and power electronics.