Multiple Neuro-Adaptive Control of Robot Manipulators Using Visual Cues

Choon-Young Lee and Ju-Jang Lee, *Senior Member, IEEE*

***Abstract*—A new adaptive controller based on multiple neural networks (NNs) for an uncertain robot manipulator system is developed in this paper. The proposed multiple neuro-adaptive controller (MNAC) switches to a memorized control skill or blends multiple skills by using visual information on the given job to improve the transient response at the time of task variation like a change of manipulating object. MNAC is a type of adaptive feedback controller where system nonlinearity terms are approximated with multiple NNs. The proposed controller is effective for a job where some tasks are repeated but information on the load cannot be scheduled before the operation. During the learning phase, MNAC memorizes a control skill for each load with each NN. For a new task, most similar existing control skills may be used as a starting point of adaptation, which improves the performance of learning. Lyapunov-function-based design of MNAC guarantees the stability of the closed-loop system to be independent of switching or blending law. Simulation results on a two-link manipulator for changing the mass of the given load were illustrated to show the effectiveness of the proposed control scheme by comparison with the conventional neuro-adaptive controller.**

***Index Terms*—Adaptive control, intelligent control, neural networks (NNs), robot manipulators, switching control.**

I. Introduction

THE robot is one of the choices for improving productivity in industrial automation. Robotic manipulators have been applied to dangerous environments and routine manufacturing jobs. As the robotic manipulators are highly nonlinear dynamic systems with uncertainty, it is difficult to obtain accurate dynamic equations for the design of control laws. Robots face many uncertainties in their dynamic models, in particular, the parameters describing the unknown grasped payloads as well as the unknown frictional coefficients. To compensate for these uncertainties, many researchers have proposed adaptive control strategies or model-free intelligent control methods.

Adaptive robot control based on the linear parameterization of robot dynamics selects a proper set of equivalent parameters such that the manipulator dynamics depend linearly on these parameters, and adaptive control algorithms can then fully account for the nonlinear time-varying nature of robot dynamics [1], [2]. Direct adaptive controllers use tracking errors of the joint motions to drive the parameter adaptation [3]. Indirect adaptive controllers use prediction errors on the filtered joint torques to generate parameter estimates to be used in the control law [4].

The application of neural network (NN) algorithms or the fuzzy logic concept for a robotic manipulator control have been interesting research topics. Because of their learning capability [5], [6], from the point of view of controller development, they have showed good performance in the robotic system control with complicated dynamic models. An effective combination of fuzzy control and a modern nonlinear control method is receiving more and more attention [7]–[11]. Artificial NNs to enhance the performance of tracking accuracy of robotic manipulators were developed with adaptation or learning algorithms for the weights of the networks [12]–[20]. Control of mechatronic servo systems by NNs was also studied by learning of the control input pattern [15].

Each study has its own advantages and disadvantages. Although all these methodologies showed good performance in controlling uncertain dynamic systems, there are unavoidable large transient errors at the time of task variation. For example, if a robot manipulator has to perform task 1, task 2, task 1, task 2, repeatedly in this order, these controllers with adaptation will adapt themselves to a newly changed task repeatedly, so the control skill having been acquired at that time will be forgotten. Although task 1 is encountered for the second time, the adaptive or learning controller recognizes the given task as a new task because the controller has already been adapted to task 2. In this case, if the dynamic parameters and control skills are stored, the stored information can be used to cope with the repeated task immediately.

Concepts for adaptive control with multiple models are not new in control theory [21], [22]. In these studies, a combination of the control determined by different models was used and no stability results were reported. We proved the overall closed-loop stability with a Lyapunov function candidate in this paper. Switching in the context of adaptive control, in which multiple models are used to determine to which controller one should switch, has been proposed [23]–[26]. The neuro-controller based on a set of fixed NNs, that is, a bank of NNs trained offline or online, with each NN representing some part of inverse dynamics was also proposed [27], [28]. In these studies, multiple NNs were used to improve the overall performance in identification and total computational requirement by using each NN covering each dynamic term. They can be thought of as a single NN in the sense that all the weights of the networks were adapted without any priority for each task and all their

Manuscript received September 5, 2002; revised December 24, 2003. Abstract published on the Internet November 10, 2004. This work was supported in part by the Human-friendly Welfare Robot System Engineering Research Center (HWRS-ERC), funded by the Korea Science and Engineering Foundation.

C.-Y. Lee was with the Digital Actor Research Team, Virtual Reality R&D Department, Electronics and Telecommunications Research Institute, Daejeon 305-350, Korea. He is now with the School of Mechanical Engineering, KyungPook National University, Daegu 702-701, Korea (e-mail: cylee7309@hanmail.net).

J.-J. Lee is with the Department of Electrical Engineering and Computer Science, Korea Advanced Institute of Science and Technology, Daejeon 305-701, Korea (e-mail: jjlee@ee.kaist.ac.kr).

Digital Object Identifier 10.1109/TIE.2004.841080

weights should be changed when system dynamics deviate from the original one. Multiple NNs, in this paper, were adopted to store different dynamics of the system for different tasks.

This paper is organized as follows. Problem formulation is described in Section II. An adaptive controller based on multiple NNs is constructed in Section III. A simulation to demonstrate the performance of the proposed method is provided in Section IV. Finally, conclusions with further study issues are given in Section V.

II. Problem Formulation

A. Dynamics of an n-Link Robot Manipulator

Consider the dynamic equation of a robot manipulator with n links

$$\boldsymbol{M}(q)\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) + \boldsymbol{F}(\dot{\boldsymbol{q}}) = \boldsymbol{\tau}(t) + \boldsymbol{d}(t) \quad (1)$$

where $\boldsymbol{q}, \dot{\boldsymbol{q}} \in \mathbb{R}^n$ are the vectors of generalized coordinates and velocities, $\boldsymbol{M}(\boldsymbol{q}) \in \mathbb{R}^{n\times n}$ the positive inertia matrix, $\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} \in \mathbb{R}^n$ the Coriolis and centrifugal torques, $\boldsymbol{G}(\boldsymbol{q}) \in \mathbb{R}^n$ the gravitational torques, $\boldsymbol{\tau}(t) \in \mathbb{R}^n$ the applied torque, and $\boldsymbol{d}(t) \in \mathbb{R}^n$ is the bounded disturbance vector representing torque disturbance. $\boldsymbol{F}(\dot{\boldsymbol{q}}) \in \mathbb{R}^n$ represents the friction term of the form

$$\boldsymbol{F}(\dot{\boldsymbol{q}}) = \boldsymbol{F}_v\dot{\boldsymbol{q}} + \boldsymbol{F}_d(\dot{\boldsymbol{q}}) \quad (2)$$

with $\boldsymbol{F}_v \in \mathbb{R}^{n\times n}$ the diagonal coefficient matrix of viscous friction, and $\boldsymbol{F}_d(\dot{\boldsymbol{q}}) \in \mathbb{R}^n$ a dynamic friction term. The following properties are required for the subsequent development.

Property 1: There exist known positive constants M_m, M_M, C_M, and G_M such that $M_m \le \|\boldsymbol{M}(\boldsymbol{q})\| \le M_M$, $\|\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\| \le C_M$, $\|\boldsymbol{G}(\boldsymbol{q})\| \le G_M$ [2].

Assumption 1: It is assumed that the norm of $\boldsymbol{d}(t)$ is bounded as

$$\|\boldsymbol{d}(t)\| \le \rho_d = \boldsymbol{\theta}_d^T\boldsymbol{\phi}_d \quad (3)$$

where $\boldsymbol{\theta}_d \in \mathbb{R}^3$ is an unknown vector, and $\boldsymbol{\phi}_d = [1, \|\boldsymbol{q}\|, \|\dot{\boldsymbol{q}}\|]^T$ is a chosen regressor vector.

The norm of a vector $\boldsymbol{x} \in \mathbb{R}^n$ and that of a matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ are defined, respectively, as $\|\boldsymbol{x}\| = \sqrt{\boldsymbol{x}^T\boldsymbol{x}}$, $\|\boldsymbol{A}\| = \sqrt{\mathrm{eig}(\boldsymbol{A}^T\boldsymbol{A})_{\max}}$, with $\mathrm{eig}(\cdot)_{\max}$ the maximum eigenvalue.

B. Approximation of Nonlinear Function Using Multiple NNs

In the field of control engineering, an NN is often used to approximate an unknown nonlinear function with a small estimation error bound. In this paper, multiple Gaussian radial basis function (RBF) NNs are considered, as shown in Fig. 1. It is a particular network architecture which uses k RBF NNs with blending coefficient $\boldsymbol{A}_i, i = 1,2,\ldots,k$. Each RBF NN uses r numbers of Gaussian function of the form

$$\phi_i(\boldsymbol{x}) = \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{c}_i\|^2}{\sigma_i^2}\right), \qquad i = 1,2,\ldots,r \quad (4)$$

where $\boldsymbol{c}_i \in \mathbb{R}^n$ is the center vector and $\sigma_i \in \mathbb{R}$ is the width of the basis function. The input vector is transformed into nonlinear functions by k RBF NNs and their outputs are multiplied by the

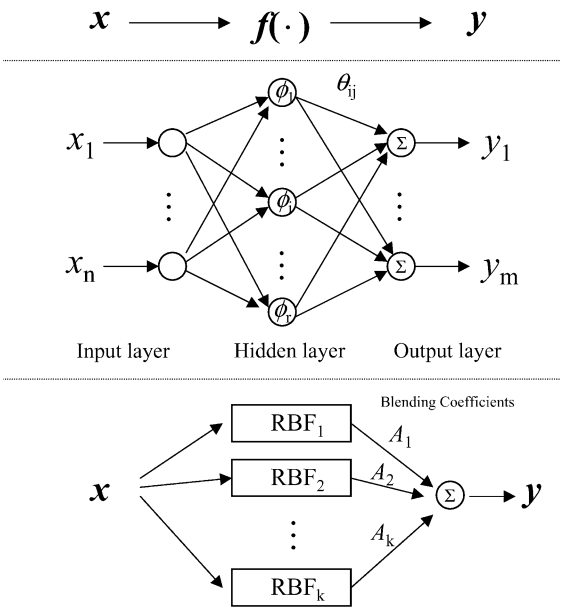


Fig. 1. Structure of multiple NNs with blending coefficient to approximate a given nonlinear function.

blending coefficients $\boldsymbol{A}_i$ to generate the resulting approximated output.

Given that $\boldsymbol{f}(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{R}^m$ is a continuous function defined on the set $\boldsymbol{x} \in \mathbb{R}^n$, and $\hat{\boldsymbol{f}}_i(\boldsymbol{\Theta}_i, \boldsymbol{x}) : \mathbb{R}^{r\times m} \times \mathbb{R}^n \to \mathbb{R}^m$ is an ith approximating function that depends continuously on the parameter matrix $\boldsymbol{\Theta}_i$ and $\boldsymbol{x}$, the approximation problem is to determine the optimal parameter $\boldsymbol{\Theta}_i^*$ such that

$$\left\|\sum_{i=1}^{k}\boldsymbol{A}_i\hat{\boldsymbol{f}}_i(\boldsymbol{\Theta}_i^*,\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{x})\right\| = \left\|\sum_{i=1}^{k}\boldsymbol{A}_i\boldsymbol{\Theta}_i^{*T}\boldsymbol{\phi}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{x})\right\| \quad (5)$$

is minimized for the given blending coefficient matrix $\boldsymbol{A}_i$, and basis functions of the network $\boldsymbol{\phi}(\boldsymbol{x})$.

III. Adaptive Controller Based on Multiple NNs

Define the useful error signals as follows:

$$\boldsymbol{e} \triangleq \boldsymbol{q} - \boldsymbol{q}_d \quad (6)$$

$$\boldsymbol{s} \triangleq \dot{\boldsymbol{e}} + \boldsymbol{\Lambda}\boldsymbol{e} \quad (7)$$

where $\boldsymbol{e} \in \mathbb{R}^n$ is the tracking error, $\boldsymbol{q}_d \in \mathbb{R}^n$ is a given twice continuously differentiable and bounded reference trajectory, $\boldsymbol{s} \in \mathbb{R}^n$ is the augmented error and $\boldsymbol{\Lambda} \in \mathbb{R}^{n\times n}$ is a constant symmetric diagonal positive definite matrix.

The dynamics for $\boldsymbol{s}$ are obtained as follows:

$$\begin{aligned}\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{s}} &= \boldsymbol{M}(\boldsymbol{q})(\ddot{\boldsymbol{e}} + \boldsymbol{\Lambda}\dot{\boldsymbol{e}}) \\ &= \boldsymbol{M}(\boldsymbol{q})(\ddot{\boldsymbol{q}} - \ddot{\boldsymbol{q}}_d + \boldsymbol{\Lambda}\dot{\boldsymbol{e}}) \\ &= \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} - \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}}_d + \boldsymbol{M}(\boldsymbol{q})\boldsymbol{\Lambda}\dot{\boldsymbol{e}} \\ &= -\boldsymbol{N}(\boldsymbol{q},\dot{\boldsymbol{q}}) - \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}}_d + \boldsymbol{M}(\boldsymbol{q})\boldsymbol{\Lambda}\dot{\boldsymbol{e}} \\ &\quad + \boldsymbol{\tau}(t) + \boldsymbol{d}(t)\end{aligned} \quad (8)$$

where $\boldsymbol{N}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) + \boldsymbol{F}(\dot{\boldsymbol{q}})$.

Consider the following Lyapunov function candidate:

$$V_1 = \frac{1}{2}\boldsymbol{s}^T\boldsymbol{M}(\boldsymbol{q})\boldsymbol{s} \tag{9}$$

to develop NNs to approximate unknown nonlinear functions. The time derivative of V_1 along the solution trajectories of the dynamics (8) is

$$\begin{aligned}\dot{V}_1 &= \frac{1}{2}\boldsymbol{s}^T\dot{\boldsymbol{M}}(\boldsymbol{q})\boldsymbol{s} + \boldsymbol{s}^T\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{s}} \\ &= \frac{1}{2}\boldsymbol{s}^T\dot{\boldsymbol{M}}(\boldsymbol{q})\boldsymbol{s} + \boldsymbol{s}^T(-\boldsymbol{N}(\boldsymbol{q},\dot{\boldsymbol{q}}) - \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}}_d \\ &\quad + \boldsymbol{M}(\boldsymbol{q})\boldsymbol{\Lambda}\dot{\boldsymbol{e}}) + \boldsymbol{s}^T\boldsymbol{\tau} + \boldsymbol{s}^T\boldsymbol{d} \\ &= \boldsymbol{s}^T\boldsymbol{\Psi}(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{q}_d,\dot{\boldsymbol{q}}_d,\ddot{\boldsymbol{q}}_d) + \boldsymbol{s}^T\boldsymbol{\tau} + \boldsymbol{s}^T\boldsymbol{d}\end{aligned} \tag{10}$$

where

$$\begin{aligned}&\boldsymbol{\Psi}(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{q}_d,\dot{\boldsymbol{q}}_d,\ddot{\boldsymbol{q}}_d) \\ &\quad = \frac{1}{2}\dot{\boldsymbol{M}}(\boldsymbol{q})\boldsymbol{s} - \boldsymbol{N}(\boldsymbol{q},\dot{\boldsymbol{q}}) - \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}}_d + \boldsymbol{M}(\boldsymbol{q})\boldsymbol{\Lambda}\dot{\boldsymbol{e}} \\ &\quad = [\Psi_1, \Psi_2, \ldots, \Psi_n]^T \in \mathbb{R}^n.\end{aligned} \tag{11}$$

The nonlinear function $\boldsymbol{\Psi}$ may be represented by multiple RBF NNs with the blending coefficient explained in Section II-B

$$\boldsymbol{\Psi} = \sum_{i=1}^{k}\boldsymbol{A}_i\boldsymbol{\Theta}_i^T\boldsymbol{\phi}(\boldsymbol{x}) + \boldsymbol{e}_a(t,\boldsymbol{x}) \tag{12}$$

where $\boldsymbol{x} \in \mathbb{R}^m, (m = 5n)$ represents the input vectors getting into the network, such as $\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{q}_d, \dot{\boldsymbol{q}}_d, \ddot{\boldsymbol{q}}_d$. $\boldsymbol{\phi}(\boldsymbol{x}) \in \mathbb{R}^r$ is the vector of r-radial basis functions chosen by the designer such that $\boldsymbol{\phi}(\boldsymbol{x})$ spans the domain of the input vector. $\boldsymbol{\Theta}_i \in \mathbb{R}^{r\times n}$ is the weight parameter matrix of the ith RBF NN. $\boldsymbol{A}_i$ represents the blending coefficient matrix for the ith RBF NN such that $\boldsymbol{A}_i \in \mathbb{R}^{n\times n}$ is a diagonal matrix whose diagonal elements are all equal to a_i, $\sum_{i=1}^{k} a_i = 1$, and $a_i \geq 0$. $\boldsymbol{e}_a(t,\boldsymbol{x}) \in \mathbb{R}^n$ is the approximation error referred to as the network reconstruction error or modeling error.

Assumption 2: There exists an unknown positive constant vector $\boldsymbol{\theta}_e \in \mathbb{R}^p$

$$\|\boldsymbol{e}_a(t,\boldsymbol{x})\| \leq \rho_e(t,\boldsymbol{x}) = \boldsymbol{\theta}_e^T\boldsymbol{\phi}_e(t,\boldsymbol{x}) \tag{13}$$

for a known positive function $\boldsymbol{\phi}_e(t,\boldsymbol{x}) \in \mathbb{R}^p$.

From the structure of the nonlinear function $\boldsymbol{\Psi}$ in (11) and (12), it can be easily shown that Assumption 2 is reasonable. We can find that the norm of the nonlinear function $\boldsymbol{\Psi}$ is bounded by a positive function from Property 1. In (12), the norm of $\sum_{i=1}^{k}\boldsymbol{A}_i\boldsymbol{\Theta}_i^T\boldsymbol{\phi}(\boldsymbol{x})$ is also bounded by a positive constant. Therefore, from Property 1, (11) and (12), $\boldsymbol{\phi}_e$ in Assumption 2 can be defined as $\boldsymbol{\phi}_e = [1, \|\ddot{\boldsymbol{q}}_d\|, \|\dot{\boldsymbol{e}}\|, \|\dot{\boldsymbol{q}}\|, \|\boldsymbol{s}\|]^T \in \mathbb{R}^p, (p = 5)$.

Theorem 1: Under Assumptions 1 and 2, if the following control law and adaptation law are applied to the robot manipulator (1), then the tracking errors converge to zero asymptotically.

Control Law:

$$\boldsymbol{\tau} = -\sum_{i=1}^{k}\boldsymbol{A}_i\hat{\boldsymbol{\Theta}}_i^T\boldsymbol{\phi}(\boldsymbol{x}) - (\hat{\rho}_d + \hat{\rho}_e)\frac{\boldsymbol{s}}{\|\boldsymbol{s}\|} - \boldsymbol{K}\boldsymbol{s} \tag{14}$$

where $\boldsymbol{K} \in \mathbb{R}^{n\times n}$ is an constant symmetric diagonal positive definite matrix, $\hat{\rho}_d$ and $\hat{\rho}_e$ are the estimates of ρ_d and ρ_e, respectively.

Adaptation Law:

$$\dot{\hat{\boldsymbol{\Theta}}}_{ij} = a_i\boldsymbol{s}_j\Gamma_1\boldsymbol{\phi}(\boldsymbol{x}), \quad i = 1,\ldots,k; \quad j = 1,\ldots,r \tag{15}$$

$$\dot{\hat{\boldsymbol{\theta}}}_d = \|\boldsymbol{s}\|\Gamma_2\boldsymbol{\phi}_d \tag{16}$$

$$\dot{\hat{\boldsymbol{\theta}}}_e = \|\boldsymbol{s}\|\Gamma_3\boldsymbol{\phi}_e \tag{17}$$

where $\hat{\boldsymbol{\Theta}}_{ij}$ is the jth column of the i-th parameter matrix, $\hat{\boldsymbol{\Theta}}_i$, $\boldsymbol{s}_j$ is the j-th element of the vector $\boldsymbol{s}$, and the adaptation gains $\Gamma_1 \in \mathbb{R}^{r\times r}, \Gamma_2 \in \mathbb{R}^{3\times 3}$, and $\Gamma_3 \in \mathbb{R}^{p\times p}$ are all positive-definite matrices.

Proof: Define a Lyapunov function candidate for the system (8)

$$\begin{aligned}V = \frac{1}{2}\boldsymbol{s}^T\boldsymbol{M}\boldsymbol{s} + \frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{k}\tilde{\boldsymbol{\Theta}}_{ij}^T\Gamma_1^{-1}\tilde{\boldsymbol{\Theta}}_{ij} \\ + \frac{1}{2}\tilde{\boldsymbol{\theta}}_d^T\Gamma_2^{-1}\tilde{\boldsymbol{\theta}}_d + \frac{1}{2}\tilde{\boldsymbol{\theta}}_e^T\Gamma_3^{-1}\tilde{\boldsymbol{\theta}}_e\end{aligned} \tag{18}$$

where $\tilde{\boldsymbol{\Theta}}_{ij} = \hat{\boldsymbol{\Theta}}_{ij} - \boldsymbol{\Theta}_{ij} \in \mathbb{R}^r$, $\tilde{\boldsymbol{\theta}}_d = \hat{\boldsymbol{\theta}}_d - \boldsymbol{\theta}_d \in \mathbb{R}^3$, and $\tilde{\boldsymbol{\theta}}_e = \hat{\boldsymbol{\theta}}_e - \boldsymbol{\theta}_e \in \mathbb{R}^p$, where $\hat{\boldsymbol{\Theta}}_{ij}, \hat{\boldsymbol{\theta}}_d$, and $\hat{\boldsymbol{\theta}}_e$ are the parameters with which corresponding nonlinear terms $\boldsymbol{\Psi}, \rho_d$ and ρ_e, respectively, are approximated.

The time derivative of V along the solution trajectories of the error dynamics (8) is found as follows:

$$\begin{aligned}\dot{V} &= \boldsymbol{s}^T\left(\sum_{i=1}^{k}\boldsymbol{A}_i\boldsymbol{\Theta}_i^T\boldsymbol{\phi}(\boldsymbol{x}) + \boldsymbol{\tau}\right) + \boldsymbol{s}^T(\boldsymbol{e}_a + \boldsymbol{d}) \\ &\quad + \sum_{j=1}^{n}\sum_{i=1}^{k}\tilde{\boldsymbol{\Theta}}_{ij}^T\Gamma_1^{-1}\dot{\hat{\boldsymbol{\Theta}}}_{ij} + \tilde{\boldsymbol{\theta}}_d^T\Gamma_2^{-1}\dot{\hat{\boldsymbol{\theta}}}_d + \tilde{\boldsymbol{\theta}}_e^T\Gamma_3^{-1}\dot{\hat{\boldsymbol{\theta}}}_e \\ &\leq \boldsymbol{s}^T\left(\sum_{i=1}^{k}\boldsymbol{A}_i\boldsymbol{\Theta}_i^T\boldsymbol{\phi}(\boldsymbol{x}) + \boldsymbol{\tau}\right) + \|\boldsymbol{s}\|(\rho_d + \rho_e) \\ &\quad + \sum_{j=1}^{n}\sum_{i=1}^{k}\tilde{\boldsymbol{\Theta}}_{ij}^T\Gamma_1^{-1}\dot{\hat{\boldsymbol{\Theta}}}_{ij} + \tilde{\boldsymbol{\theta}}_d^T\Gamma_2^{-1}\dot{\hat{\boldsymbol{\theta}}}_d + \tilde{\boldsymbol{\theta}}_e^T\Gamma_3^{-1}\dot{\hat{\boldsymbol{\theta}}}_e.\end{aligned} \tag{19}$$

By applying the control law (14), we obtain

$$\begin{aligned}\dot{V} &\leq \boldsymbol{s}^T\left(\sum_{i=1}^{k}\boldsymbol{A}_i\tilde{\boldsymbol{\Theta}}_i^T\boldsymbol{\phi}(\boldsymbol{x}) - (\hat{\rho}_d + \hat{\rho}_e)\frac{\boldsymbol{s}}{\|\boldsymbol{s}\|} - \boldsymbol{K}\boldsymbol{s}\right) \\ &\quad + \|\boldsymbol{s}\|(\rho_d + \rho_e) \\ &\quad + \sum_{j=1}^{n}\sum_{i=1}^{k}\tilde{\boldsymbol{\Theta}}_{ij}^T\Gamma_1^{-1}\dot{\hat{\boldsymbol{\Theta}}}_{ij} + \tilde{\boldsymbol{\theta}}_d^T\Gamma_2^{-1}\dot{\hat{\boldsymbol{\theta}}}_d + \tilde{\boldsymbol{\theta}}_e^T\Gamma_3^{-1}\dot{\hat{\boldsymbol{\theta}}}_e \\ &\leq -\boldsymbol{s}^T\boldsymbol{K}\boldsymbol{s} - \boldsymbol{s}^T\sum_{i=1}^{k}\boldsymbol{A}_i\tilde{\boldsymbol{\Theta}}_i^T\boldsymbol{\phi}(\boldsymbol{x}) - (\tilde{\rho}_d + \tilde{\rho}_e)\|\boldsymbol{s}\| \\ &\quad + \sum_{j=1}^{n}\sum_{i=1}^{k}\tilde{\boldsymbol{\Theta}}_{ij}^T\Gamma_1^{-1}\dot{\hat{\boldsymbol{\Theta}}}_{ij} + \tilde{\boldsymbol{\theta}}_d^T\Gamma_2^{-1}\dot{\hat{\boldsymbol{\theta}}}_d + \tilde{\boldsymbol{\theta}}_e^T\Gamma_3^{-1}\dot{\hat{\boldsymbol{\theta}}}_e\end{aligned} \tag{20}$$

where $\tilde{\rho}_d = \tilde{\boldsymbol{\theta}}_d^T\boldsymbol{\phi}_d$ and $\tilde{\rho}_e = \tilde{\boldsymbol{\theta}}_e^T\boldsymbol{\phi}_e$.

Rearranging the above equation using $\dot{\tilde{\Theta}} = \dot{\hat{\Theta}}, \dot{\tilde{\theta}}_d = \dot{\hat{\theta}}_d$, and $\dot{\tilde{\theta}}_e = \dot{\hat{\theta}}_e$,

$$\dot{V} \le -\boldsymbol{s}^T \boldsymbol{K} \boldsymbol{s} + \sum_{i=1}^{k}\sum_{j=1}^{n} \tilde{\Theta}_{ij}^T \left(\Gamma_1^{-1} \dot{\hat{\Theta}}_{ij} - a_i \boldsymbol{s}_j \boldsymbol{\phi}(\boldsymbol{x}) \right) + \tilde{\boldsymbol{\theta}}_d^T \left(\Gamma_2^{-1} \dot{\hat{\boldsymbol{\theta}}}_d - \boldsymbol{\phi}_d \|\boldsymbol{s}\| \right) + \tilde{\boldsymbol{\theta}}_e^T \left(\Gamma_3^{-1} \dot{\hat{\boldsymbol{\theta}}}_e - \boldsymbol{\phi}_e \|\boldsymbol{s}\| \right) \quad (21)$$

where $\boldsymbol{s}_j$ is the jth element of the vector $\boldsymbol{s}$. If we apply the adaptation law of (15)–(17), we can finally obtain

$$\dot{V} \le -\boldsymbol{s}^T \boldsymbol{K} \boldsymbol{s}. \quad (22)$$

This implies that $V(t) \le V(0)$ and, therefore, that $\boldsymbol{s}, \tilde{\Theta}, \tilde{\boldsymbol{\theta}}_d$ and $\tilde{\boldsymbol{\theta}}_e$ are all bounded. Accordingly, from (14), $\boldsymbol{\tau}$ is also bounded. Since $\boldsymbol{s}$ is bounded, $\boldsymbol{e}$ and $\dot{\boldsymbol{e}}$ are also bounded according to the stable dynamics $\boldsymbol{s} = \dot{\boldsymbol{e}} + \boldsymbol{\Lambda} \boldsymbol{e}$. $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$ are bounded since the desired trajectories $\boldsymbol{q}_d, \dot{\boldsymbol{q}}_d$ and $\ddot{\boldsymbol{q}}_d$ are all bounded. Consequently, from (8), $\dot{\boldsymbol{s}}$ is also bounded.

To use Barbalat's lemma [2], let us check the uniform continuity of $\dot{V}$. The derivative of $\dot{V}$ is

$$\ddot{V} \le -2\boldsymbol{s}^T \boldsymbol{K} \dot{\boldsymbol{s}}. \quad (23)$$

This shows that $\ddot{V}$ is bounded and, hence, $\dot{V}$ is uniformly continuous. Application of Barbalat's lemma then indicates that $\dot{V}$ converges to zero as time goes to infinity. Therefore, $\boldsymbol{s}$ is globally asymptotically stable and, hence, the tracking errors $\boldsymbol{e}$ and $\dot{\boldsymbol{e}}$ also converge to zero globally and asymptotically by the stable dynamics of $\boldsymbol{s}$. ■

To alleviate the chattering of the control input caused by the discontinuous control law of (14), we can introduce a modified controller as follows:

$$\boldsymbol{\tau} = -\sum_{i=1}^{k} \boldsymbol{A}_i \hat{\boldsymbol{\Theta}}_i^T \boldsymbol{\phi}(\boldsymbol{x}) - (\hat{\rho}_d + \hat{\rho}_e) \frac{\boldsymbol{s}}{\|\boldsymbol{s}\| + \epsilon} - \boldsymbol{K} \boldsymbol{s} \quad (24)$$

where ϵ is a small positive constant.

To inhibit infinite growing of $\hat{\boldsymbol{\theta}}_d$ and $\hat{\boldsymbol{\theta}}_e$, we can use the following adaptation law:

$$\dot{\hat{\boldsymbol{\theta}}}_d = \Gamma_2 (\|\boldsymbol{s}\| \boldsymbol{\phi}_d - \bar{\sigma}_d \hat{\boldsymbol{\theta}}_d) \quad (25)$$

$$\dot{\hat{\boldsymbol{\theta}}}_e = \Gamma_3 (\|\boldsymbol{s}\| \boldsymbol{\phi}_e - \bar{\sigma}_e \hat{\boldsymbol{\theta}}_e) \quad (26)$$

where $\bar{\sigma}_d$ and $\bar{\sigma}_e$ are the σ-modification terms. This modification overcomes the problem of continuous increase in the parameter estimation. If the above continuous controller (24) and adaptation law (15), (25), and (26) are applied to the system, then it is a well-known fact that the tracking errors are globally uniformly ultimately bounded.

From Theorem 1, blending between multiple models does not affect the closed-loop stability, that is, we can design any blending rule for appropriate operation. In this paper, we assumed that we can recognize an object on the end-effector by a vision system as in the concept of a system configuration (Fig. 2). We can register some objects on a vision system before operation, and we can check which object is now operated by the robot manipulator. If an unknown object is applied, the vision system selects the most similar object among stored data, and then, the controller uses the corresponding network for the selected object. In this manner, we can switch between multiple models using the vision system. Another possible choice of blending rule for the unknown object is to use a visual similarity measure between the current object and the stored object [29].

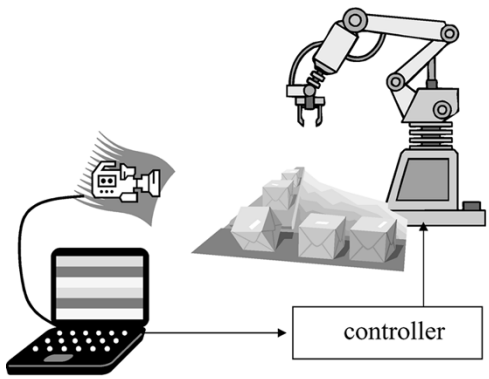


Fig. 2. Concept of system configuration.

IV. Simulation

In this section, we will apply our proposed control scheme to control a two-link robot manipulator system to track a given trajectory with changing mass of the load. We will compare the performance of multiple neuro-adaptive controller (MNAC) with that of a conventional neuro-adaptive controller (NAC) for achieving the given task.

The dynamic equations of the two-link manipulator system are

$$\begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} + \begin{bmatrix} -h\dot{q}_2 & -h\dot{q}_1 - h\dot{q}_2 \\ h\dot{q}_1 & 0 \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} + \begin{bmatrix} g_1 \\ g_2 \end{bmatrix} + \begin{bmatrix} v_1 \dot{q}_1 \\ v_2 \dot{q}_2 \end{bmatrix} + \begin{bmatrix} k_1 \mathrm{sgn}(\dot{q}_1) \\ k_2 \mathrm{sgn}(\dot{q}_2) \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} \quad (27)$$

where

$$\begin{aligned} M_{11} &= I_1 + I_2 + m_1 l_{c1}^2 + m_2 \left(l_1^2 + l_{c2}^2 + 2 l_1 l_{c2} \cos(q_2) \right) \\ &\quad + m_3 \left(l_1^2 + l_2^2 + 2 l_1 l_2 \cos(q_2) \right) \\ M_{12} &= I_2 + m_2 \left(l_{c2}^2 + l_1 l_{c2} \cos(q_2) \right) \\ &\quad + m_3 \left(l_2^2 + l_1 l_2 \cos(q_2) \right) \\ M_{21} &= M_{12} \\ M_{22} &= I_2 + m_2 l_{c2}^2 + m_3 l_2^2 \\ h &= m_2 l_1 l_{c2} \sin(q_2) \\ g_1 &= m_1 l_{c1} g \cos(q_1) \\ &\quad + m_2 g (l_{c2} \cos(q_1 + q_2) + l_1 \cos(q_1)) \\ g_2 &= m_2 l_{c2} g \cos(q_1 + q_2). \end{aligned}$$

The matrix $\boldsymbol{M}$ can be shown to be positive definite and, therefore, always invertible. In our simulation, we use the following parameter values: $m_1 = 1.0$ kg, mass of link one; $m_2 = 1.0$ kg, mass of link two; $l_1 = 1.0$ m, length of link one; $l_2 = 1.0$ m, length of link two; $l_{c1} = 0.5$ m, distance from the joint of link one to its center of gravity (COG); $l_{c2} = 0.5$ m, distance from the joint of link two to its COG; $I_1 = 0.2$ kg·m^2, lengthwise centroidal inertia of link one; $I_2 = 0.2$ kg·m^2, lengthwise centroidal inertia of link two; $g = 9.8$ m/s^2; $v_1 = v_2 = 0.1$,

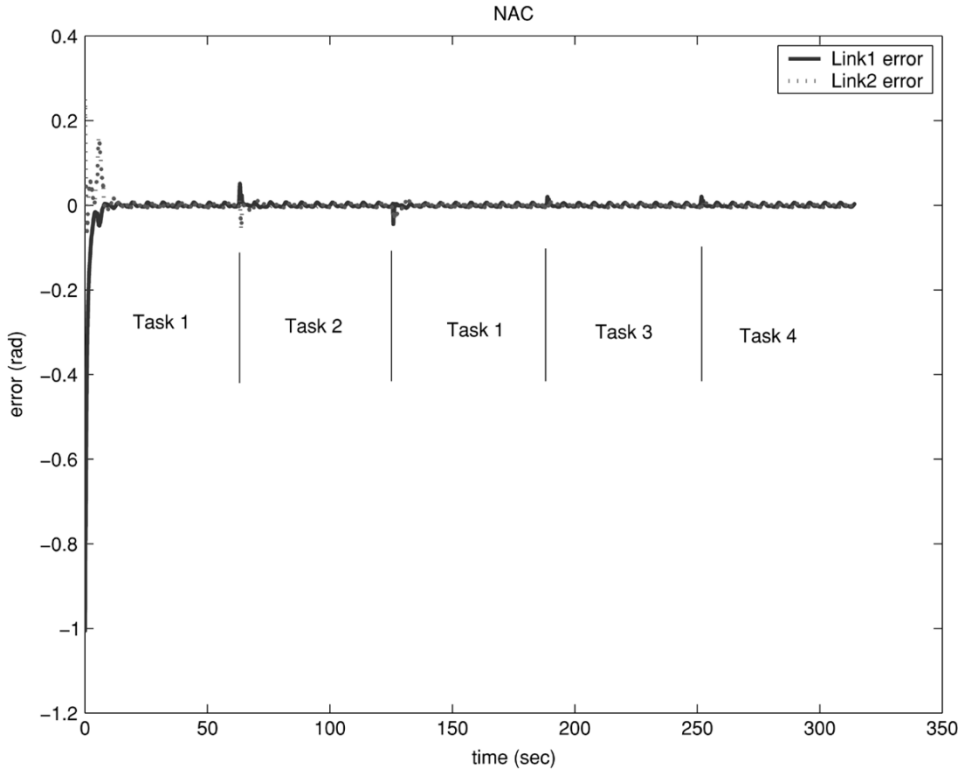


Fig. 3. Tracking error for desired trajectories (NAC).

coefficient of viscous friction; $k_1 = k_2 = 0.1$, coefficient of dynamic friction. The mass m_3 represents the mass of an object at the end of the link.

The control objective is to control the state $\boldsymbol{q}$ of the system to track the reference trajectory $\boldsymbol{q}_d = [\cos(t)\ \sin(t)]^T$, and during the operation, the mass of an object at the end of the link changes as follows:

$$m_3 = \begin{cases} 0.0\text{ kg}, & \text{if } 0 \le t \le t_1 \quad \text{task 1} \\ 3.0\text{ kg}, & \text{if } t_1 < t \le t_2 \quad \text{task 2} \\ 0.0\text{ kg}, & \text{if } t_2 < t \le t_3 \quad \text{task 1} \\ 1.0\text{ kg}, & \text{if } t_3 < t \le t_4 \quad \text{task 3} \\ 2.0\text{ kg}, & \text{if } t_4 < t \le t_5 \quad \text{task 4} \end{cases}$$

where $t_1 = 10\pi$, $t_2 = 20\pi$, $t_3 = 30\pi$, $t_4 = 40\pi$, and $t_5 = 50\pi$ second. Task 1, task 2, and task 3 are assumed to be for the known objects before operation and task 4 is assumed to be for the unknown object.

Since there are three known objects, we designed three RBF networks. We placed 121 basis functions evenly on the region of controllability. We used adaptation gains as $\Gamma_1 = 10.0I$, and $\Gamma_2 = \Gamma_3 = 0.1I$ through the simulation where I is an identity matrix. We set the external disturbance as $\boldsymbol{d} = [0.1\sin(10t)\ 0.1\cos(10t)]^T$. We calculated $\boldsymbol{s}$ using

$$\boldsymbol{\Lambda} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \text{and} \quad \boldsymbol{K} = \begin{bmatrix} 15 & 0 \\ 0 & 15 \end{bmatrix}$$

was used in the controller equation.

Two groups of simulations were performed: 1) conventional NAC which has a single RBF network and 2) the proposed MNAC which has three RBF networks.

Fig. 3 shows the tracking errors when NAC is applied. Initially, an NN does not have dynamic knowledge of the system and adaptive schemes change network weights when task 1 is given for $0 \le t \le t_1$. As time goes by, the performance is improved since NN approximates nonlinearity of the system for the task 1. When task 2 is given for $t_1 < t \le t_2$, system dynamics for this task are different from that of the learned NN at the time since the mass of the load has changed. Therefore, some transient errors appear at around $t = t_1$ and the NN starts to be adapted for this new task minimizing tracking errors. At $t = t_2$, the task has changed into the old task 1, there are also some transient errors around at that time as shown in Fig. 3, since the NN forgot the control skill for task 1 while it learned a new control skill for task 2. A conventional NAC repeats its adaptation for the given task whether it is a new task or an old task since it has no memory for storing the control skill. At every change of task, there show transient errors. If there are large task variations, the conventional NAC will give large transient errors when tasks change.

Fig. 4 shows the tracking errors when MNAC is applied. For $0 \le t \le t_2$, the performance is similar to that of NAC. Since tasks 1, 2, and 3 are known, three NNs cover these tasks. The control skill for each task is stored for each NN and an appropriate control skill can be selected for the repeated task. As shown in Fig. 4, there are small transient errors at around $t = t_2$ when the task has changed into the old task 1 of which dynamic information has been memorized with one of the multiple NNs. For the unknown new task 4, the control skill for the known object which is most similar to this new unknown object will be selected and it becomes the starting point of adaptation to this new task. Therefore, adaptation time can be reduced and control

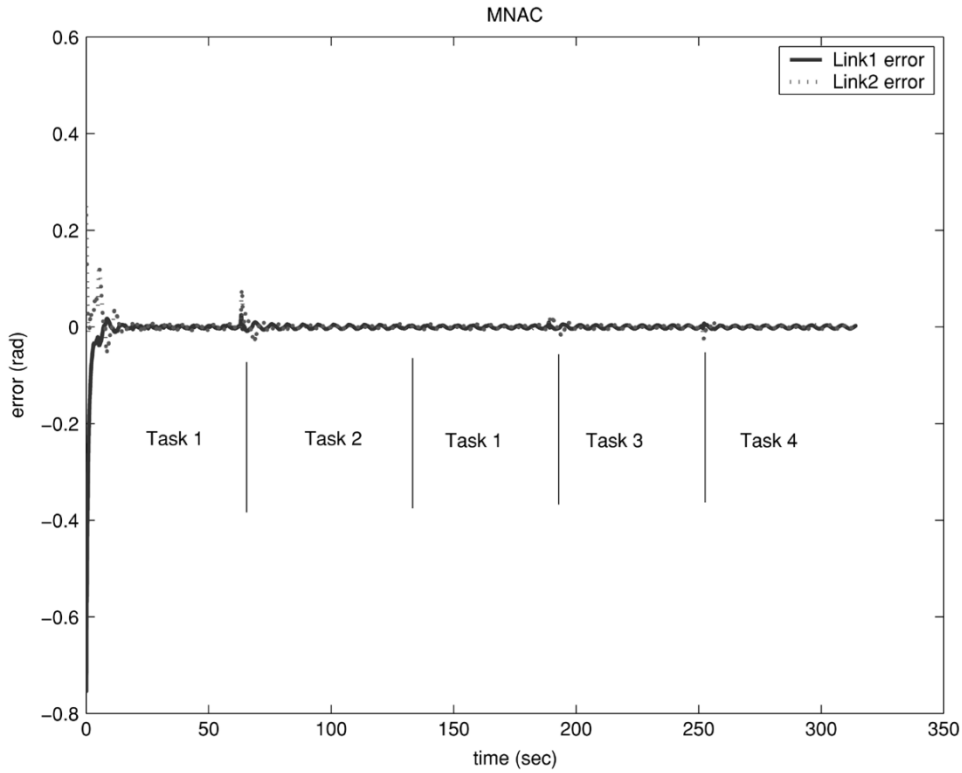


Fig. 4. Tracking error for desired trajectories (MNAC).

TABLE I
PERFORMANCE COMPARISON BETWEEN NAC AND MNAC

Interval	IAE for NAC	IAE for MNAC	$\left(\frac{\text{IAE}_{\text{NAC}}-\text{IAE}_{\text{MNAC}}}{\text{IAE}_{\text{NAC}}}\right)\times 100\%$
$t\in[0,t_5]$	3.3568	3.0608	8.8171 %
$t\in[t_2,t_3]$	0.5300	0.3154	40.5031 %
$t\in[t_3,t_4]$	0.3823	0.3572	6.5667 %
$t\in[t_4,t_5]$	0.3454	0.2971	13.9753 %

performance can also be slightly improved. From these simulations, the proposed MNAC shows better performance over conventional NAC for the repeated tasks which is needed in industrial application where load changes from time to time.

We compared the control performance for the two groups of simulations (ten trials for each controller) by averaging the integrals of the absolute magnitude of the error, IAE, which is written as

$$\text{IAE}=\int_{T_0}^{T_1}|\boldsymbol{e}(t)|\,dt. \tag{28}$$

Table I presents the comparison of control performance in IAE between NAC and MNAC. The integral of absolute errors for each interval are listed for the two controllers. Through the whole simulation time, MNAC improved IAE over NAC by about 9%. However, during the interval $t\in[t_2,t_3]$, where task 1 is revisited, MNAC improved IAE over NAC by about 40%, which is very significant. Also, for the unknown task 4, $t\in[t_4,t_5]$, IAE for MNAC is slightly improved. The results show that the proposed MNAC will be effective in tracking performance when some tasks are repeated.

Although MNAC has considerably more neurons than NAC, the reason for performance improvement by MNAC is mainly due to the memory of control skills using independent multiple RBF networks. To achieve similar improvements, a system with one RBF network (NAC) should have a learning algorithm of memorizing control skills without affecting the previous memory. MNAC used multiple RBF networks as a solution for memorizing a new control skill without losing previous memory.

V. CONCLUSION

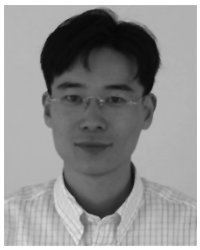
In this paper, a multiple neuro-adaptive control method was developed for the control of a robotic manipulator in picking up and placing various objects. Multiple NNs are used to approximate the changing system dynamics for various tasks. For the case of repeated jobs, the proposed control scheme is effective because of its capability of memorizing a control skill for each task with the designed NN. Transient errors at the time of change of tasks will be attenuated significantly when a task occurs repeatedly as shown in the simulation. Lyapunov-function-based design of adaptation laws guarantees the global stability of the closed-loop system.

The online generation and pruning of multiple models will be interesting further topics. A fuzzy-inference-based blending rule for multiple models with visual information is currently being studied.

REFERENCES

[1] S. S. Sastry and A. Isidori, "Adaptive control of linearization systems," *IEEE Trans. Autom. Control*, vol. 34, no. 11, pp. 1123–1131, Nov. 1989.
[2] F. L. Lewis, C. T. Abdallah, and D. M. Dawson, *Control of Robot Manipulators*. New York: Macmillan, 1993.

[3] J.-J. E. Slotine and W. Li, "On the adaptive control of robot manipulators," *Int. J. Robot. Res.*, vol. 6, pp. 49–59, 1987.
[4] R. H. Middleton and G. C. Goodwin, "Adaptive computed torque control for rigid link manipulators," *Syst. Control Lett.*, vol. 10, pp. 9–16, 1988.
[5] K. Funahashi, "On the approximate realization of continuous mapping by neural networks," *Neural Netw.*, vol. 2, no. 3, pp. 359–366, 1989.
[6] J. L. Castro, "Fuzzy logical controllers are universal approximators," *IEEE Trans. Syst., Man, Cybern.*, vol. 25, no. 4, pp. 629–635, Apr. 1995.
[7] L. X. Wang, "Stable adaptive fuzzy control of nonlinear systems," *IEEE Trans. Fuzzy Syst.*, vol. 1, no. 2, pp. 146–155, May 1993.
[8] J. T. Spooner and K. M. Passino, "Stable adaptive control using fuzzy systems and neural networks," *IEEE Trans. Fuzzy Syst.*, vol. 4, no. 3, pp. 339–359, Aug. 1996.
[9] M. R. Emmi, A. A. Goldenberg, and I. B. Turksen, "Fuzzy logic dynamics modeling of robot manipulators," in *Proc. IEEE Conf. Robotics and Automation*, Leuven, Belgium, 1998, pp. 2512–2517.
[10] M. C. M. Teixeira and S. H. Zak, "Stabilizing controller design for uncertain nonlinear systems using fuzzy model," *IEEE Trans. Fuzzy Syst.*, vol. 7, no. 2, pp. 133–142, Apr. 1999.
[11] C.-H. Wang, H.-L. Liu, and T.-C. Lin, "Direct adaptive fuzzy-neural control with state observer and supervisory controller for unknown nonlinear dynamical systems," *IEEE Trans. Fuzzy Syst.*, vol. 10, no. 1, pp. 39–49, Feb. 2002.
[12] J. Albus, "A new approach to manipulator control: The cerebella model articulation controller (CMAC)," *Trans. ASME, J. Dyn. Syst., Meas., Control*, vol. 63, no. 3, pp. 220–227, 1975.
[13] R. M. Sanner and J.-J. E. Slotine, "Gaussian networks for direct adaptive control," *IEEE Trans. Neural Netw.*, vol. 3, no. 6, pp. 837–863, Nov. 1992.
[14] R. Carelli, E. F. Camacho, and D. Patiño, "A neural network based feedforward adaptive controller for robot," *IEEE Trans. Syst., Man, Cybern*, vol. 25, no. 9, pp. 1281–1288, Sep. 1995.
[15] S. Goto, M. Nakamura, and N. Kyura, "Accurate contour control of mechatronic servo systems using Gaussian networks," *IEEE Trans. Ind. Electron.*, vol. 43, no. 4, pp. 469–476, Aug. 1996.
[16] G. A. Rovithakis and M. A. Christodulou, "Neural adaptive regulation of unknown nonlinear dynamical systems," *IEEE Trans. Syst., Man, Cybern. B*, vol. 27, no. 5, pp. 810–822, Oct. 1997.
[17] Y. G. Leu, T. T. Lee, and W. Y. Wang, "Observer-based adaptive fuzzy neural control for unknown nonlinear dynamical systems," *IEEE Trans. Syst., Man, Cybern.*, vol. 29, no. 5, pp. 583–591, Oct. 1999.
[18] F. L. Lewis, S. Jagannathan, and A. Yesildirek, *Neurl Network Control of Robot Manipulators and Nonlinear Systems*, London, U.K.: Taylor & Francis, 1999.
[19] S. Seshagiri and H. K. Khalil, "Output feedback control of nonlinear systems using RBF neural networks," *IEEE Trans. Neural Netw.*, vol. 11, no. 1, pp. 69–79, Jan. 2000.
[20] F. Sun, Z. Sun, and P.-Y. Woo, "Neural network-based adaptive controller design of robotic manipulators with an observer," *IEEE Trans. Neural Netw.*, vol. 12, no. 1, pp. 54–67, Jan. 2001.
[21] M. Athans *et al.*, "The stochastic control of the F-8C aircraft using a multiple model adaptive control (MMAC) method—Part I: Equilibrium flight," *IEEE Trans. Autom. Control*, vol. 22, no. 5, pp. 768–780, Oct., 1977.
[22] C. Yu, R. J. Roy, H. Kaufman, and B. W. Bequette, "Multiple-model adaptive predictive control of mean arterial pressure and cardiac output," *IEEE Trans. Bio-Med. Eng.*, vol. 39, no. 8, pp. 765–778, Aug. 1992.
[23] M. Fu and B. R. Barmish, "Adaptive stabilization of linear systems via switching control," *IEEE Trans. Autom. Control*, vol. 31, no. 12, pp. 1097–1103, Dec. 1986.
[24] A. S. Morse, D. Q. Mayne, and G. C. Goodwin, "Applications of hysteresis switching in parameter adaptive control," *IEEE Trans. Autom. Control*, vol. 37, no. 9, pp. 1343–1354, Sep. 1992.
[25] K. S. Narendra and J. Balakrishnan, "Adaptive control using multiple models," *IEEE Trans. Autom. Control*, vol. 42, no. 2, pp. 171–187, Feb. 1997.
[26] D. J. Leith and W. E. Leithead, "Analytic framework for blended multiple model systems using linear local models," *Int. J. Control*, vol. 72, pp. 605–619, 1999.
[27] D. Shukla, D. M. Dawson, and F. W. Paul, "Multiple neural-network-based adaptive controller using orthonormal activation function neural networks," *IEEE Trans. Neural Netw.*, vol. 10, no. 6, pp. 1494–1501, Nov. 1999.
[28] H. D. Patiño, R. Carelli, and B. R. Kuchen, "Neural networks for advanced control of robot manipulators," *IEEE Trans. Neural Netw.*, vol. 13, no. 2, pp. 343–354, mar. 2002.
[29] S. Meikle and R. Yates, "Computer vision algorithms for autonomous mobile robot map bluilding and path palnning," in *Proc. Conf. Systems Sciences*, vol. 3, 1998, pp. 292–301.



Choon-Young Lee was born in Korea in 1973. He received the B.S. degree in electronic engineering from Hanyang University, Seoul, Korea, in 1996, and the M.S. degree and the Ph.D. degree in electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 1998 and 2003, respectively.

From 2003 to 2004, he was a Senior Researcher with the Digital Actor Research Team, Electronics and Telecommunications Research Institute (ETRI), Daejeon, Korea. In 2005, he joined the School of Mechanical Engineering, KyungPook National University, Daegu, Korea, where he is currently a Lecturer. From 1998 to 2000, he developed a gait training robot system as a Researcher on a project sponsored by the Ministry of Health and Welfare, Korea. He also participated in the development of an emotional robot from 2001 to 2002. His research interests are in the fields of neural network applications for the design of control system, ITS, sensor fusion, and robotics.

Dr. Lee is a Member of the Korean Society of Automotive Engineers and the Institute of Control, Automation, Systems Engineers.

Ju-Jang Lee (M'86–SM'98) was born in Seoul, Korea, in 1948. He received the B.S. and M.S. degrees from Seoul National University, Seoul, Korea, in 1973 and 1977, respectively, and the Ph.D. degree in electrical engineering from the University of Wisconsin, Madison, in 1984, all in electrical engineering.

From 1977 to 1978, he was a Research Engineer with the Korean Electric Research and Testing Institute, Seoul, Korea. From 1978 to 1979, he was a Design and Processing Engineer with G. T. E. Automatic Electric Company, Waukesha, WI. In 1983, he was briefly the Project Engineer with the Research and Development Department of the Wisconsin Electric Power Company, Milwaukee, WI. In 1984, he joined the Department of Electrical Engineering, Korea Advanced Institute of Science and Technology (KAIST), Daejon, Korea, where he is currently a Professor. In 1987, he was a Visiting Professor at the Robotics Laboratory of Imperial College of Science and Technology, London, U.K. From 1991 to 1992, he was a Visiting Scientist at the Robotics Institute at Carnegie Mellon University, Pittsburgh, PA. His research interests are in the areas of intelligent control of mobile robots, service robotics for the disabled, space robotics, evolutionary computation, variable-structure control, chaotic control systems, electronic control units for automobiles, and power system stabilizers.

Dr. Lee is a Member of the IEEE Robotics and Automation, IEEE Evolutionary Computation, and IEEE Industrial Electronics Societies, the Korean Institute of Electrical Engineers, the Institute of Electronics Engineers of Korea, and the Korea Information Science Society. He is a Vice President of the ICASE and a Director of the Society of Instrument and Control Engineers of Japan.