The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Multiple Incremental Fuzzy Neuro-Adaptive Control of Robot Manipulators

Chang-Hyun Kim, Joon-Hong Seok, Byoung-Suk Choi, and Ju-Jang Lee, *Fellow, IEEE*

Abstract— **An adaptive control using multiple incremental fuzzy neural networks (FNNs) is proposed for robot manipulators. The structure and parameters of the FNNs are determined dynamically by using an incremental FNN. By incorporating incremental learning and adaptive control with multiple models, the proposed method not only reduces complexity and computation induced by the use of multiple models, but also provides favorable transient and tracking performance. The multiple FNNs are switched or blended to improve the transient response when manipulating objects are changed. The parameters are refined adaptively to compensate for system uncertainties. The resulting closed-loop system with a switching or blending law is proven to be asymptotically stable. The proposed scheme is applied to control a two-link robot manipulator in conjunction with varying payloads.**

I. INTRODUCTION

Robot manipulators are essential components for improving product quality and productivity in automated factories. Because, robot manipulators have complex nonlinear dynamics with uncertainties such as payload, friction, and disturbances, it is difficult or even impossible to construct an accurate robot model. Therefore, much attention has been paid to model-free control methods such as neural networks and fuzzy systems.

As for robot control, many adaptive control strategies with neural networks (NNs) or fuzzy systems have been presented in the literature [1], [2]. Most of these strategies use FNNs to approximate nonlinearity such as the inverse dynamics of the manipulator and unknown disturbances. However, these FNN based control methods all require predefined and fixed fuzzy rules or a NN structure, which is difficult to determine in advance due to the tradeoff between the structure and accuracy. Numerous rules or neurons may be required to cover all possible input domains, especially in a case such as robots where the dimension of the input is high, which results in redundant or inefficient computation.

Several self-organizing FNNs have been applied to the control problem [3], [4]. Fuzzy rules are built up incrementally, starting with none at the beginning. Incremental algorithms can run efficiently, but their performance may depend on the data presentation order. Most self-organizing FNNs explicitly or implicitly utilize some sort of hierarchical learning switching from initial coarse learning to fine learning. After convergence of learning, FNNs have problems coping with changing environments due to the discrepancy between the previously learned rule and a new rule.

The authors are with the Department of Electrical Engineering, Korea Advanced Institute of Science and Technology, 305-701 Daejeon, Korea (e-mail: sunnine@odyssey.kaist.ac.kr).

Task variation such as replacing tools or manipulating different objects is common in industrial fields. In this case, switching control from one task to another is useful. Although FNN based methods show good performance in robot control problems, there are unavoidable transient errors at the time of task variation. Multiple models are used for this purpose. Switching of multiple adaptive models has been proposed [5]–[7]. Model reference adaptive control or neuro-controllers based on a set of fixed NNs are used to deal with each model. For the neuro-controllers, the NNs are trained offline and the switching parameters are adapted online. Therefore, they have disadvantages such as difficulty of structure assignment, necessity of many neurons, and a high computation load, as mentioned earlier. Moreover, the burden is increased because multiple models are stored. Online generation and pruning of multiple models are necessary, which has yet to be addressed in the literature.

In this paper, an adaptive control scheme using multiple incremental FNNs is proposed. The overall controller is comprised of a feedback controller and multiple FNNs that learn inverse dynamics of the robot manipulator for different tasks. The multiple FNNs are generated dynamically using an incremental hyperplane-based fuzzy clustering algorithm [8]. To compensate for unknown disturbances of the system, the parameters are adapted online and a robust controller with adaptive bound estimation is included. Asymptotical stability of the closed-loop system with a switching or blending of multiple FNNs is established using the Lyapunov theory. By incorporating incremental learning and adaptive control with multiple models, the proposed method not only reduces complexity and computation induced by the use of multiple models, but also provides favorable transient and tracking performance.

II. PRELIMINARIES

A. Dynamics of an m-Link Robot Manipulator

The dynamic equation of an m-link robot manipulator can be expressed in the following Lagrange form:

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

where $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}} \in \mathbb{R}^m$ are the vectors of generalized coordinates, velocities, accelerations, $\boldsymbol{M}(\boldsymbol{q}) \in \mathbb{R}^{m \times m}$ the positive inertia matrix, $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} \in \mathbb{R}^m$ the Coriolis and centrifugal torques, $\boldsymbol{G}(\boldsymbol{q}) \in \mathbb{R}^m$ the gravitational torques, $\boldsymbol{\tau}(t) \in \mathbb{R}^m$ the applied torque, and $\boldsymbol{d}(t) \in \mathbb{R}^m$ is bounded disturbance vector representing torque disturbance. The following properties of the robot manipulator are well-known.

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

Property 2: $\dot{\boldsymbol{M}}(\boldsymbol{q}) - 2\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})$ is skew-symmetric.

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

$$\|\boldsymbol{d}(t)\| \le \rho_d = \boldsymbol{w}_d^T \boldsymbol{\phi}_d \tag{2}$$

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

It is noted that the friction term, which depends on $\dot{\boldsymbol{q}}$, is assumed to be included in $\boldsymbol{d}(t)$.

B. Nonlinear Function Approximation with Multiple FNNs

FNNs are used to approximate a nonlinear function. Especially, Takagi-Sugeno-Kang (TSK) fuzzy models can provide a suitable framework for function approximation [9]. The rule consequents are represented as linear functions of the model inputs, while the antecedent part specifies the operational region of a rule.

Let $\boldsymbol{x} = [x_1, \dots, x_n]^T \in \mathbb{R}^n$ be the input variable and $\boldsymbol{y} \in \mathbb{R}^m$ be the output variable. A symmetric Gaussian membership function (MF) and the weighted sum defuzzification are used. For any given input vector $\boldsymbol{x}$, the output can be written in the following matrix form:

$$\boldsymbol{y} = \begin{bmatrix} \boldsymbol{w}_1^T \boldsymbol{\psi}_1(\boldsymbol{x}) \\ \boldsymbol{w}_2^T \boldsymbol{\psi}_2(\boldsymbol{x}) \\ \vdots \\ \boldsymbol{w}_m^T \boldsymbol{\psi}_m(\boldsymbol{x}) \end{bmatrix} = \boldsymbol{W}^T \boldsymbol{\Psi}(\boldsymbol{x}) \tag{3}$$

where $\boldsymbol{w}_j$ and $\boldsymbol{\psi}_j(\boldsymbol{x})$ are the consequent parameters and the regressors of the jth output with appropriate dimensions. The number of rules and parameters (mean, deviation, and linear weights) are defined differently for each output.

Next, multiple FNNs are considered. Let there exist N FNNs. The output of each FNN is multiplied by the blending coefficient α_i to get the overall output as

$$\boldsymbol{y} = \sum_{i=1}^{N} \alpha_i \boldsymbol{W}_i \boldsymbol{\Psi}_i(\boldsymbol{x}). \tag{4}$$

The blending coefficients α_i are scalar values which satisfy $\sum_{i=1}^{N} \alpha_i = 1$ and $\alpha_i \ge 0$. These coefficients enable us to combine all FNNs or to switch from one FNN to another.

III. Multiple Incremental Fuzzy Neuro-Adaptive Control

In this section, an adaptive controller with multiple FNNs that learn inverse dynamics of the robot manipulator for different tasks is presented. In the proposed method, any self-organizing modeling method can be used for this purpose. An incremental hyperplane-based fuzzy clustering (IHFC) [8] is adopted in this research because it is adequate for system modeling and generates interpretable fuzzy rules. Some requisite relations are introduced first.

For the given trajectory, the error signal is defined as

$$\boldsymbol{e} \triangleq \boldsymbol{q} - \boldsymbol{q}_d \tag{5}$$

where $\boldsymbol{q}_d \in \mathbb{R}^m$ is a twice differentiable and bounded desired trajectory. And define the filtered tracking error as

$$\boldsymbol{s} \triangleq \dot{\boldsymbol{e}} + \lambda \boldsymbol{e} \tag{6}$$

where λ is a scalar design parameter.

The dynamics of $\boldsymbol{s}$ may be written as

$$\begin{aligned} \boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{s}} &= \boldsymbol{M}(\boldsymbol{q})(\ddot{\boldsymbol{q}} - \ddot{\boldsymbol{q}}_d + \lambda \dot{\boldsymbol{e}}) \\ &= -\boldsymbol{M}(\boldsymbol{q})(\ddot{\boldsymbol{q}}_d - \lambda\dot{\boldsymbol{e}}) - \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{G}(\boldsymbol{q}) + \boldsymbol{\tau} + \boldsymbol{d} \\ &= -\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\boldsymbol{s} - \boldsymbol{M}(\boldsymbol{q})(\ddot{\boldsymbol{q}}_d - \lambda\dot{\boldsymbol{e}}) \\ &\quad - \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})(\dot{\boldsymbol{q}}_d - \lambda\boldsymbol{e}) - \boldsymbol{G}(\boldsymbol{q}) + \boldsymbol{\tau} + \boldsymbol{d} \\ &= -\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\boldsymbol{s} - \boldsymbol{f}_d - \Delta w + \boldsymbol{\tau} + \boldsymbol{d} \end{aligned} \tag{7}$$

where $\boldsymbol{f}_d$ and Δw are defined as follows

$$\boldsymbol{f}_d = \boldsymbol{M}(\boldsymbol{q}_d)\ddot{\boldsymbol{q}}_d + \boldsymbol{C}(\boldsymbol{q}_d,\dot{\boldsymbol{q}}_d)\dot{\boldsymbol{q}}_d + \boldsymbol{G}(\boldsymbol{q}_d) \tag{8}$$

$$\Delta w = \boldsymbol{M}(\boldsymbol{q})(\ddot{\boldsymbol{q}}_d - \lambda\dot{\boldsymbol{e}}) + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})(\dot{\boldsymbol{q}}_d - \lambda\boldsymbol{e}) + \boldsymbol{G}(\boldsymbol{q}) - \boldsymbol{f}_d. \tag{9}$$

The computed feedforward (or inverse robot dynamics) term $\boldsymbol{f}_d$ may be represented by multiple FNNs as

$$\boldsymbol{f}_d = \sum_{i=1}^{N} \alpha_i \boldsymbol{W}_i \boldsymbol{\Psi}_i(\boldsymbol{x}) + \boldsymbol{e}_a \tag{10}$$

where $\boldsymbol{x} \in \mathbb{R}^n, (n = 3m)$ is the input vector of the FNNs, which consists of $\boldsymbol{q}_d$, $\dot{\boldsymbol{q}}_d$ and $\ddot{\boldsymbol{q}}_d$. $\boldsymbol{e}_a \in \mathbb{R}^m$ is the approximation error referred to as the network reconstruction error or modeling error.

Assumption 2: It is assumed that $\boldsymbol{e}_a$ is bounded as

$$\|\boldsymbol{e}_a\| \le \rho_e = \boldsymbol{w}_e^T \boldsymbol{\phi}_e \tag{11}$$

where $\boldsymbol{w}_e$ is unknown vector, and $\boldsymbol{\phi}_e \in \mathbb{R}^p$ is a known positive function.

From the universal approximation property [9], Assumption 2 is reasonable. We can find that the norm of $\boldsymbol{f}_d$ in (8) is bounded by a positive function from Property 1. From the linear relation in the output and accompanying equations, the norm of $\sum_{i=1}^{N} \alpha_i \boldsymbol{W}_i \boldsymbol{\Psi}_i(\boldsymbol{x})$ is bounded by a positive constant and $\|\boldsymbol{x}\|$. Therefore, $\boldsymbol{\phi}_e$ in Assumption 2 can be defined as $\boldsymbol{\phi}_e = [1, \|\boldsymbol{q}_d\|, \|\dot{\boldsymbol{q}}_d\|, \|\ddot{\boldsymbol{q}}_d\|]^T \in \mathbb{R}^4, (p = 4)$.

Multiple FNNs are generated dynamically using IHFC to estimate $\boldsymbol{f}_d$. IHFC determines the structure and parameters of the FNN, which is characterized by $\boldsymbol{W}_i$ and $\boldsymbol{\Psi}_i$. After the initial value of the weight $\boldsymbol{W}_{i0}$ is obtained from the IHFC algorithm, $\boldsymbol{W}_i$ is further adjusted online to compensate for modeling errors of the multiple FNNs. The IHFC learning operates in discrete time at every sampling time and the adaptation of the weight $\boldsymbol{W}_i$ occurs in continuous time. Therefore, $\boldsymbol{W}_i$ and $\boldsymbol{\Psi}_i$ may vary in dimension concurrently to reflect the structure of the multiple FNNs during the control process. Assumption 2 can be fulfilled with various choices of $\boldsymbol{W}_i$ and $\boldsymbol{\Psi}_i$ and we further assume that IHFC can always find one of these choices. In the IHFC algorithm, if the modeling error becomes large, a new rule is created, and as a consequence a different choice is selected to reduce the modeling error. Also, the structure of $\boldsymbol{\Psi}_i$ is expected to vary slightly during the process of online adaptation, since

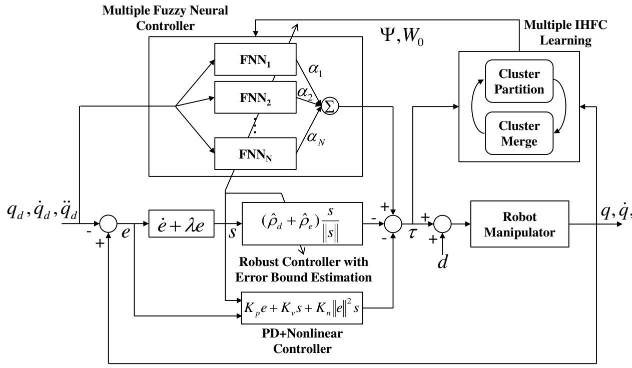


Fig. 1. Multiple incremental fuzzy neural control architecture.

adaptation is performed along the bounded desired trajectory until the next sampling time. Thus, we fix $\boldsymbol{\Psi}_i$ and only $\boldsymbol{W}_i$ is further adapted to reduce the error. Therefore, the error becomes smaller and this assumption seems feasible.

Finally, the adaptive controller in the following theorem is proposed. The PD controller and the nonlinear controller guarantee asymptotic stability during the learning phase. In addition, we include the robust term with the bound estimation to remove the effects of unknown disturbances and modeling error. Fig. 1 depicts the architecture of the proposed controller.

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

Control Law:

$$\boldsymbol{\tau} = \sum_{i=1}^{N} \alpha_i \hat{\boldsymbol{W}}_i \boldsymbol{\Psi}_i(\boldsymbol{x}) - (\hat{\rho}_d + \hat{\rho}_e)\frac{\boldsymbol{s}}{\|\boldsymbol{s}\|} - \boldsymbol{K}_n \|\boldsymbol{e}\|^2 \boldsymbol{s} - \boldsymbol{K}_p \boldsymbol{e} - \boldsymbol{K}_v \boldsymbol{s} \tag{12}$$

where $\boldsymbol{K}_n = k_n\boldsymbol{I}$, $\boldsymbol{K}_p = k_p\boldsymbol{I}$ and $\boldsymbol{K}_v = k_v\boldsymbol{I}$ are sufficiently large constant diagonal positive definite gain matrices (k_n, k_p, $k_v > 0$), $\hat{\boldsymbol{W}}_i$, $\hat{\rho}_d$ and $\hat{\rho}_e$ are the estimates of $\boldsymbol{W}_i$, ρ_d and ρ_e, respectively.

Adaptation Law:

$$\dot{\hat{\boldsymbol{W}}}_{ij} = -\alpha_i s_j \boldsymbol{\Gamma}_{1j} \boldsymbol{\Psi}_{ij}, \quad i = 1, \ldots, N; \quad j = 1, \ldots, m \tag{13}$$

$$\dot{\hat{\boldsymbol{w}}}_d = \|\boldsymbol{s}\| \boldsymbol{\Gamma}_2 \boldsymbol{\phi}_d \tag{14}$$

$$\dot{\hat{\boldsymbol{w}}}_e = \|\boldsymbol{s}\| \boldsymbol{\Gamma}_3 \boldsymbol{\phi}_e \tag{15}$$

where $\hat{\boldsymbol{W}}_{ij}$ and $\boldsymbol{\Psi}_{ij}$ is the estimated consequent parameters and the regressors corresponding the jth output of ith FNN's parameter matrix given in (3), $\boldsymbol{s}_j$ is the jth element of the vector $\boldsymbol{s}$, $\hat{\boldsymbol{w}}_d$ and $\hat{\boldsymbol{w}}_e$ are the estimated parameters of $\boldsymbol{w}_d$ and $\boldsymbol{w}_e$, and the adaptation gains $\boldsymbol{\Gamma}_{1j}$, $\boldsymbol{\Gamma}_2$ and $\boldsymbol{\Gamma}_3$ are all positive definite matrices with appropriate dimensions.

Proof: Consider the following Lyapunov function

$$V = \frac{1}{2}\boldsymbol{s}^T\boldsymbol{M}(\boldsymbol{q})\boldsymbol{s} + \frac{1}{2}\boldsymbol{e}^T\boldsymbol{K}_p\boldsymbol{e} + \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{m}\tilde{\boldsymbol{W}}_{ij}^T\boldsymbol{\Gamma}_{1j}^{-1}\tilde{\boldsymbol{W}}_{ij} + \frac{1}{2}\tilde{\boldsymbol{w}}_d^T\boldsymbol{\Gamma}_2^{-1}\tilde{\boldsymbol{w}}_d + \frac{1}{2}\tilde{\boldsymbol{w}}_e^T\boldsymbol{\Gamma}_3^{-1}\tilde{\boldsymbol{w}}_e \tag{16}$$

where $\tilde{\boldsymbol{W}}_{ij} = \hat{\boldsymbol{W}}_{ij} - \boldsymbol{W}_{ij}$, $\tilde{\boldsymbol{w}}_d = \hat{\boldsymbol{w}}_d - \boldsymbol{w}_d$, and $\tilde{\boldsymbol{w}}_e = \hat{\boldsymbol{w}}_e - \boldsymbol{w}_e$.

Then, the time derivative of V is obtained by using (7) and (10) as

$$\begin{aligned}
\dot{V} &= \frac{1}{2}\boldsymbol{s}^T\dot{\boldsymbol{M}}(\boldsymbol{q})\boldsymbol{s} + \boldsymbol{s}^T\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{s}} + \boldsymbol{e}^T\boldsymbol{K}_p\dot{\boldsymbol{e}} \\
&\quad + \sum_{i=1}^{N}\sum_{j=1}^{m}\tilde{\boldsymbol{W}}_{ij}^T\boldsymbol{\Gamma}_{1j}^{-1}\dot{\tilde{\boldsymbol{W}}}_{ij} + \tilde{\boldsymbol{w}}_d^T\boldsymbol{\Gamma}_2^{-1}\dot{\tilde{\boldsymbol{w}}}_d + \tilde{\boldsymbol{w}}_e^T\boldsymbol{\Gamma}_3^{-1}\dot{\tilde{\boldsymbol{w}}}_e \\
&= \frac{1}{2}\boldsymbol{s}^T\dot{\boldsymbol{M}}(\boldsymbol{q})\boldsymbol{s} - \boldsymbol{s}^T\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\boldsymbol{s} + \boldsymbol{s}^T(-\boldsymbol{f}_d - \Delta w + \boldsymbol{\tau} + \boldsymbol{d}) \\
&\quad + \sum_{i=1}^{N}\sum_{j=1}^{m}\tilde{\boldsymbol{W}}_{ij}^T\boldsymbol{\Gamma}_{1j}^{-1}\dot{\tilde{\boldsymbol{W}}}_{ij} + \tilde{\boldsymbol{w}}_d^T\boldsymbol{\Gamma}_2^{-1}\dot{\tilde{\boldsymbol{w}}}_d + \tilde{\boldsymbol{w}}_e^T\boldsymbol{\Gamma}_3^{-1}\dot{\tilde{\boldsymbol{w}}}_e \\
&\quad + \boldsymbol{e}^T\boldsymbol{K}_p\dot{\boldsymbol{e}} \\
&= \boldsymbol{s}^T(-\sum_{i=1}^{N}\alpha_i\boldsymbol{W}_i\boldsymbol{\Psi}_i(\boldsymbol{x}) - \boldsymbol{e}_a - \Delta w + \boldsymbol{\tau} + \boldsymbol{d}) \\
&\quad + \sum_{i=1}^{N}\sum_{j=1}^{m}\tilde{\boldsymbol{W}}_{ij}^T\boldsymbol{\Gamma}_{1j}^{-1}\dot{\tilde{\boldsymbol{W}}}_{ij} + \tilde{\boldsymbol{w}}_d^T\boldsymbol{\Gamma}_2^{-1}\dot{\tilde{\boldsymbol{w}}}_d + \tilde{\boldsymbol{w}}_e^T\boldsymbol{\Gamma}_3^{-1}\dot{\tilde{\boldsymbol{w}}}_e \\
&\quad + \boldsymbol{e}^T\boldsymbol{K}_p\dot{\boldsymbol{e}} \\
&\le \boldsymbol{s}^T(-\sum_{i=1}^{N}\alpha_i\boldsymbol{W}_i\boldsymbol{\Psi}_i(\boldsymbol{x}) + \boldsymbol{\tau}) + \|\boldsymbol{s}\|(\rho_d + \rho_e) \\
&\quad + \sum_{i=1}^{N}\sum_{j=1}^{m}\tilde{\boldsymbol{W}}_{ij}^T\boldsymbol{\Gamma}_{1j}^{-1}\dot{\tilde{\boldsymbol{W}}}_{ij} + \tilde{\boldsymbol{w}}_d^T\boldsymbol{\Gamma}_2^{-1}\dot{\tilde{\boldsymbol{w}}}_d + \tilde{\boldsymbol{w}}_e^T\boldsymbol{\Gamma}_3^{-1}\dot{\tilde{\boldsymbol{w}}}_e \\
&\quad + \boldsymbol{e}^T\boldsymbol{K}_p\dot{\boldsymbol{e}} - \boldsymbol{s}^T\Delta w
\end{aligned} \tag{17}$$

where the skew-symmetric property is used.

By applying the control law (12), then we obtain

$$\begin{aligned}
\dot{V} &\le \boldsymbol{s}^T(\sum_{i=1}^{N}\alpha_i\tilde{\boldsymbol{W}}_i\boldsymbol{\Psi}_i(\boldsymbol{x}) - (\hat{\rho}_d + \hat{\rho}_e)\frac{\boldsymbol{s}}{\|\boldsymbol{s}\|} \\
&\quad - \boldsymbol{K}_n\|\boldsymbol{e}\|^2\boldsymbol{s} - \boldsymbol{K}_p\boldsymbol{e} - \boldsymbol{K}_v\boldsymbol{s}) + \|\boldsymbol{s}\|(\rho_d + \rho_e) \\
&\quad + \sum_{i=1}^{N}\sum_{j=1}^{m}\tilde{\boldsymbol{W}}_{ij}^T\boldsymbol{\Gamma}_{1j}^{-1}\dot{\tilde{\boldsymbol{W}}}_{ij} + \tilde{\boldsymbol{w}}_d^T\boldsymbol{\Gamma}_2^{-1}\dot{\tilde{\boldsymbol{w}}}_d + \tilde{\boldsymbol{w}}_e^T\boldsymbol{\Gamma}_3^{-1}\dot{\tilde{\boldsymbol{w}}}_e \\
&\quad + \boldsymbol{e}^T\boldsymbol{K}_p\dot{\boldsymbol{e}} - \boldsymbol{s}^T\Delta w \\
&= \boldsymbol{s}^T\sum_{i=1}^{N}\alpha_i\tilde{\boldsymbol{W}}_i\boldsymbol{\Psi}_i(\boldsymbol{x}) - (\tilde{\rho}_d + \tilde{\rho}_e)\|\boldsymbol{s}\| \\
&\quad + \sum_{i=1}^{N}\sum_{j=1}^{m}\tilde{\boldsymbol{W}}_{ij}^T\boldsymbol{\Gamma}_{1j}^{-1}\dot{\tilde{\boldsymbol{W}}}_{ij} + \tilde{\boldsymbol{w}}_d^T\boldsymbol{\Gamma}_2^{-1}\dot{\tilde{\boldsymbol{w}}}_d + \tilde{\boldsymbol{w}}_e^T\boldsymbol{\Gamma}_3^{-1}\dot{\tilde{\boldsymbol{w}}}_e \\
&\quad + k_p\boldsymbol{e}^T\dot{\boldsymbol{e}} - k_n\|\boldsymbol{e}\|^2\boldsymbol{s}^T\boldsymbol{s} - k_p\boldsymbol{s}^T\boldsymbol{e} - k_v\boldsymbol{s}^T\boldsymbol{s} - \boldsymbol{s}^T\Delta w
\end{aligned} \tag{18}$$

where $\tilde{\rho}_d = \tilde{\boldsymbol{w}}_d^T\boldsymbol{\phi}_d$ and $\tilde{\rho}_e = \tilde{\boldsymbol{w}}_e^T\boldsymbol{\phi}_e$. Substituting $\dot{\tilde{\boldsymbol{W}}}_{ij} = \dot{\hat{\boldsymbol{W}}}_{ij}$, $\dot{\tilde{\boldsymbol{w}}}_d = \dot{\hat{\boldsymbol{w}}}_d$, and $\dot{\tilde{\boldsymbol{w}}}_e = \dot{\hat{\boldsymbol{w}}}_e$ yields

$$\begin{aligned}
\dot{V} &\le \sum_{i=1}^{N}\sum_{j=1}^{m}\tilde{\boldsymbol{W}}_{ij}^T(\boldsymbol{\Gamma}_{1j}^{-1}\dot{\hat{\boldsymbol{W}}}_{ij} + \alpha_i\boldsymbol{s}_j\boldsymbol{\Psi}_{ij}) \\
&\quad + \tilde{\boldsymbol{w}}_d^T(\boldsymbol{\Gamma}_2^{-1}\dot{\hat{\boldsymbol{w}}}_d - \boldsymbol{\phi}_d\|\boldsymbol{s}\|) + \tilde{\boldsymbol{w}}_e^T(\boldsymbol{\Gamma}_3^{-1}\dot{\hat{\boldsymbol{w}}}_e - \boldsymbol{\phi}_e\|\boldsymbol{s}\|) \\
&\quad + k_p\boldsymbol{e}^T\dot{\boldsymbol{e}} - k_n\|\boldsymbol{e}\|^2\boldsymbol{s}^T\boldsymbol{s} - k_p\boldsymbol{s}^T\boldsymbol{e} - k_v\boldsymbol{s}^T\boldsymbol{s} - \boldsymbol{s}^T\Delta w
\end{aligned}$$

(19)

where the summation is divided by each element. If the adaptation law of (13)–(15) is applied, the above equation is written as

$$\dot{V} \le k_p \boldsymbol{e}^T \dot{\boldsymbol{e}} - k_n \|\boldsymbol{e}\|^2 \boldsymbol{s}^T\boldsymbol{s} - k_p \boldsymbol{s}^T \boldsymbol{e} - k_v \boldsymbol{s}^T \boldsymbol{s} - \boldsymbol{s}^T \Delta w. \quad (20)$$

We can use the result in [10] to obtain the upper bound on Δw by

$$\|\Delta w\| \le b_1 \|\boldsymbol{e}\| + b_2 \|\boldsymbol{e}\|^2 + b_3 \|s\| + b_4 \|\boldsymbol{s}\| \|\boldsymbol{e}\| \quad (21)$$

where b_1, b_2, b_3, and b_4 are positive bounds which depend on the desired trajectory and the physical properties of the manipulator. By using (6) and (21), a new upper bound on $\dot{V}$ can be obtained as

$$\begin{aligned}\dot{V} \le & -k_p\lambda \|\boldsymbol{e}\|^2 - k_n \|\boldsymbol{e}\|^2 \|\boldsymbol{s}\|^2 - k_v \|\boldsymbol{s}\|^2 + \|\boldsymbol{s}\| \|\Delta w\| \\ \le & -k_p\lambda \|\boldsymbol{e}\|^2 - k_n \|\boldsymbol{e}\|^2 \|\boldsymbol{s}\|^2 - k_v \|\boldsymbol{s}\|^2 + b_1 \|\boldsymbol{s}\| \|\boldsymbol{e}\| \\ & - b_2 \|\boldsymbol{e}\|^2 [1/2 - \|\boldsymbol{s}\|]^2 - b_4 \|\boldsymbol{s}\|^2 [1/2 - \|\boldsymbol{e}\|]^2 \\ & + (b_2 + b_4) \|\boldsymbol{e}\|^2 \|\boldsymbol{s}\|^2 + (b_2/4) \|\boldsymbol{s}\|^2 + (b_3 + b_4/4) \|\boldsymbol{s}\|^2 \\ = & -(k_p\lambda - b_2/4) \|\boldsymbol{e}\|^2 - (k_v - b_3 - b_4/4) \|\boldsymbol{s}\|^2 \\ & + b_1 \|\boldsymbol{s}\| \|\boldsymbol{e}\| - (k_n - b_2 - b_4) \|\boldsymbol{e}\|^2 \|\boldsymbol{s}\|^2 \\ & - b_2 \|\boldsymbol{e}\|^2 [1/2 - \|\boldsymbol{s}\|]^2 - b_4 \|\boldsymbol{s}\|^2 [1/2 - \|\boldsymbol{e}\|]^2. \end{aligned} \quad (22)$$

If the control gain k_n is chosen as

$$k_n > b_2 + b_4 \quad (23)$$

and the other non-positive terms on the last line of (22) are eliminated, we obtain

$$\dot{V} \le -(k_p\lambda - b_2/4) \|\boldsymbol{e}\|^2 - (k_v - b_3 - b_4/4) \|\boldsymbol{s}\|^2 + b_1 \|\boldsymbol{s}\| \|\boldsymbol{e}\| \quad (24)$$

This equation can be written in the matrix form:

$$\dot{V} \le -[\|\boldsymbol{e}\| \, \|\boldsymbol{s}\|] \boldsymbol{Q} \begin{bmatrix} \|\boldsymbol{e}\| \\ \|\boldsymbol{s}\| \end{bmatrix} \quad (25)$$

where

$$\boldsymbol{Q} = \begin{bmatrix} k_p\lambda - b_2/4 & -b_1/2 \\ -b_1/2 & k_v - b_3 - b_4/4 \end{bmatrix}. \quad (26)$$

The application of Gerschgorin theorem gives the sufficient conditions on k_p and k_v that ensures the positive definite $\boldsymbol{Q}$ as

$$k_p > (b_1/2 + b_2/4)/\lambda \quad (27)$$

and

$$k_v > b_1/2 + b_3 + b_4/4. \quad (28)$$

Therefore, from (25), $\boldsymbol{s}$ and $\boldsymbol{e}$ are asymptotically stable and, hence, $\dot{\boldsymbol{e}}$ is also asymptotically stable ($\boldsymbol{e}$, $\dot{\boldsymbol{e}} \to 0$ as $t \to \infty$). ■

To prevent the chattering phenomenon of the control input, we can modify the robust control law as:

$$-(\hat{\rho}_d + \hat{\rho}_e)\frac{\boldsymbol{s}}{\|\boldsymbol{s}\| + \varepsilon} \quad (29)$$

where ε is a small positive constant. However, we can show the ultimate uniform boundedness of the system instead of the asymptotical stability in this case. Furthermore, σ-modification or e-modification laws can be used to inhibit infinite growing of $\hat{\boldsymbol{w}}_d$ and $\hat{\boldsymbol{w}}_e$.

From Theorem 1, blending between multiple models does not affect the closed-loop stability, and several strategies can be used to determine the blending coefficients effectively. Some objects are registered and the change of object can be recognized by various sensors such as tactile, force, and vision sensors. For known tasks, one of the blending coefficients is set to 1 and the fuzzy models can be switched simply. For unknown tasks, the blending coefficients can be selected to simply switch to the closest fuzzy model or to mix several sets of fuzzy models. Automatic switching using several criteria based on the estimation error [5], [6] is another possible alternative.

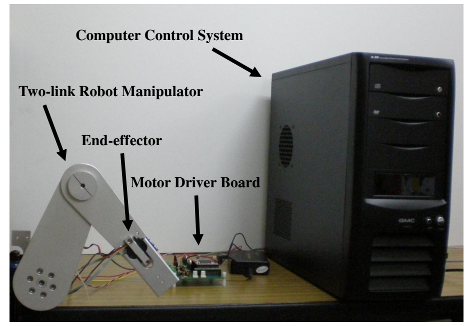


Fig. 2. Image of the robot manipulator and the control system for the experiment.

IV. Experimental results

Some experiments have been carried out for a two-link robot manipulator where the load mass is varied. We compare the proposed multiple incremental fuzzy neuro-adaptive controller (MIFNAC) with a conventional PID controller, an adaptive controller [11], and a multiple neuro-adaptive controller (MNAC) [7]. MNAC has NNs with predefined fixed Gaussian radial basis functions (RBFs). It uses quite similar control and adaptation laws to MIFNAC except for the nonlinear terms and the structure of the NN. We also include single model versions (NAC and IFNAC, respectively) of MNAC and MIFNAC in the comparison to show the advantages of using the multiple models.

The used manipulator and its control system are depicted in Fig. 2. Two dc motors are used and a TMS320F2812 digital signal processor (DSP) is used to interface with the encoder and driver circuits. The control interval is chosen to be 2 ms, and the IHFC learning operates at 100 ms.

The desired trajectories are given as $\boldsymbol{q}_d = [\cos(\pi t), \sin(\pi t)]^T$. $\lambda = 1$ is used and all initial conditions of the robot axes are set to zero.

During operation, the mass of the load changes as follows:

$$m_3 = \begin{cases} 0.0 \text{ kg}, & \text{if } 0 \le t \le t_1 \quad \text{Task 1} \\ 0.5 \text{ kg}, & \text{if } t_1 < t \le t_2 \quad \text{Task 2} \\ 0.0 \text{ kg}, & \text{if } t_2 < t \le t_3 \quad \text{Task 1} \\ 0.2 \text{ kg}, & \text{if } t_3 < t \le t_4 \quad \text{Task 3} \\ 0.35 \text{ kg}, & \text{if } t_4 < t \le t_5 \quad \text{Task 4} \end{cases}$$

where $t_1 = 9, t_2 = 19, t_3 = 29, t_4 = 39$ and $t_5 = 50$ second. For convenience, the robot manipulator changes objects when the end-effector of the robot reaches its lowest position. The objects for Task 1, Task 2, and Task 3 are assumed to be known but the object for Task 4 is not. Therefore, we prepare 3 multiple models for the experiment. In this experiment, the blending coefficients are determined in advance considering the task variation. Task 4 is similar to both Task 2 and Task 3, and thus the blending coefficients of Task 2 and Task 3 are set to 0.5.

PID gains are selected as $\boldsymbol{K}_p = 250\boldsymbol{I}$ and $\boldsymbol{K}_i = \boldsymbol{K}_d = 50\boldsymbol{I}$. The adaptive controller uses control gain $\boldsymbol{K}_v = 7\boldsymbol{I}$ and adaptation gain $\boldsymbol{\Gamma} = 5.0\boldsymbol{I}$. The parameters of NAC and MNAC are as follows: $\boldsymbol{\Gamma}_{1j} = 5.0\boldsymbol{I}, j = 1, 2, \ldots, m$, $\boldsymbol{\Gamma}_2 = \boldsymbol{\Gamma}_3 = 0.005\boldsymbol{I}$, and $\boldsymbol{K}_v = 7\boldsymbol{I}$. Also, 121 basis functions are assigned evenly on the region of controllability. For incremental rule generation of IFNAC and MIFNAC, the parameters of IHFC are chosen as $[\sigma_I, \epsilon, \tau] = [0.2, 0.1, 0.05]$. The adaptation and control gains are selected as follows: $\boldsymbol{\Gamma}_{1j} = 5.0\boldsymbol{I}, j = 1, 2, \ldots, m$, $\boldsymbol{\Gamma}_2 = \boldsymbol{\Gamma}_3 = 0.005\boldsymbol{I}$, and $\boldsymbol{K}_p = \boldsymbol{K}_v = \boldsymbol{K}_n = 3.5\boldsymbol{I}$.

Fig. 3 and Fig 4 show the tracking errors for link 1 and link 2, respectively. With the elapse of time, adaptation occurs and the tracking error is reduced. The PID controller achieves the best rising time by applying large torque, but gives poor tracking performance. The adaptive controller gives the worst transient response and similar steady-state tracking performance to the PID controller. NAC (MNAC) and IFNAC (MIFNAC) give a similar transient response, but IFNAC (MIFNAC) provides a better steady-state tracking error.

At $t = t_1$, the task is changed to Task 2 and thus the system dynamics also changes for $t_1 < t \leq t_2$. The performance of the PID controller does not improve, since it does not use any knowledge about the robot. Other adaptive controllers again start their adaptation to learn the dynamics and some transient errors appear. In this case, MIFNAC can give less tracking error after learning is finished, because supervised IHFC learning is used to approximate the inverse dynamics. However, IFNAC shows residual steady state error, resulting from the single fuzzy model having already learned the dynamics of Task 1. The IHFC learning hardly adapts to the new environment after the learning is finished as other self-organizing FNNs do.

For $t_2 < t \leq t_3$, the task is changed to the former Task 1 again and multiple models prove their strengths here. The single model approaches (adaptive controller and NAC) have to be adapted to Task 1 again, since it forgot the control skill for Task 1 while it learned a new control skill for Task 2. Therefore, some transient errors due to the task transition are also seen. On the other hand, small transient errors can be found with multiple models (MNAC and MIFNAC) that have the control skill for Task 1 as one of the multiple models. In addition, IFNAC also gives small tracking error because it has the previously learned memory of Task 1.

For $t_3 < t \leq t_4$, the task is changed to Task 3. All controllers except for the PID controller again start their adaptation, and similar descriptions can be made for $t_1 < t \leq t_2$. Finally, the task is changed to unknown Task 4 for $t_4 < t \leq t_5$ and the control skills for the known tasks that are similar to this unknown task will be used. The control skills for Task 2 and Task 3 are blended for Task 4. By using these control skills, the adaptation time and the transient error due to task transition can be reduced.

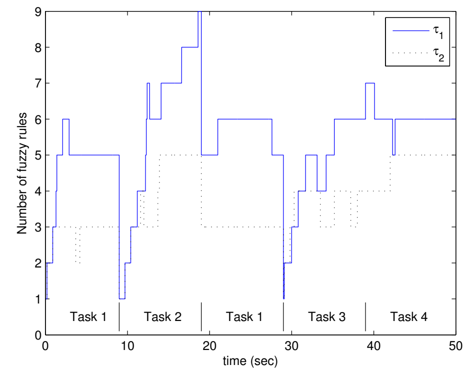


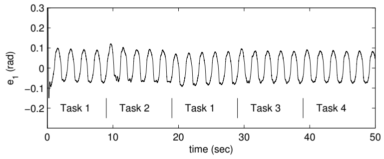
Fig. 5. Number of fuzzy rules.

Fig. 5 illustrates the generation of fuzzy rules of MIFNAC. At the beginning of learning when the task transitions occur, the fuzzy rules are generated and merged dynamically. After the learning is converged, 3–9 rules are sufficient to fulfill the desired tasks, which is very parsimonious in comparison with non-incremental methods (NAC and MNAC).

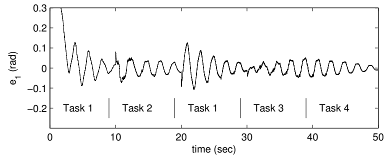
We compared the control performance by averaging the integrals of the absolute magnitude of the error (IAE). Table I summarizes the comparison results. The tracking performance of NAC is improved slightly compared with the adaptive controller. Multiple model approaches provide better tracking performance relative to other controllers, especially their single versions. Through the whole experiment time, MIFNAC gives superior IAE over other methods. In particular, during the interval $t \in [t_2, t_3]$, where the former Task 1 is revisited, multiple models improved IAE significantly and MIFNAC improved IAE even more. While IFNAC can give quite good IAE during this interval, the overall performance is not good, because it showed biased behaviors in the tracking control of other tasks. For the unknown Task 4, $t \in [t_4, t_5]$, IAE for MIFNAC is improved slightly.

V. Conclusion

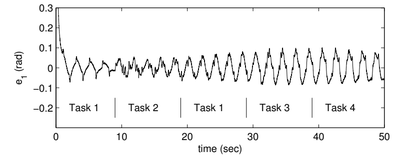
In this paper, an adaptive control scheme for robot manipulators using multiple incremental FNNs is proposed. The multiple FNNs are used to approximate the changing system dynamics for various tasks. The multiple FNNs are generated dynamically by using incremental hyperplane-based fuzzy clustering. Although multiple models use more neurons or rules than a single model in general, the incremental scheme relaxes this requirement considerably, since the rule has its own MFs and only necessary rules are generated. For repeated jobs, the proposed control scheme reduced transient errors due to task transition effectively. Also, the incremental



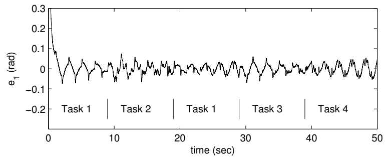
(a) PID controller.



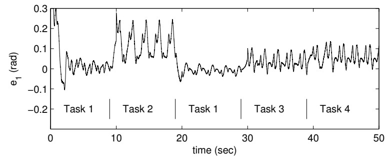
(b) Adaptive controller.



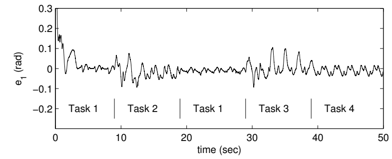
(c) NAC.



(d) MNAC.

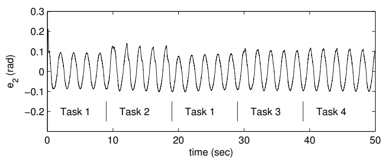


(e) IFNAC.

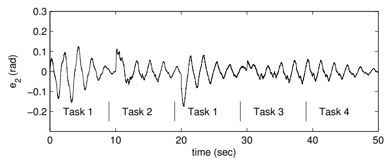


(f) MIFNAC.

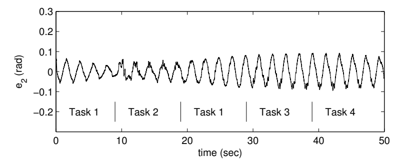
Fig. 3. Experimental tracking results of link 1.



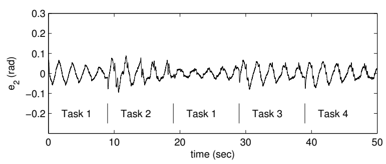
(a) PID controller.



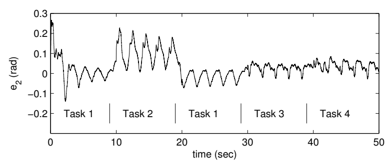
(b) Adaptive controller.



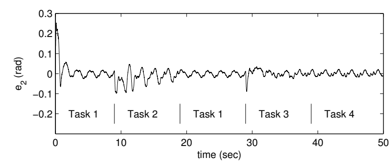
(c) NAC.



(d) MNAC.



(e) IFNAC.



(f) MIFNAC.

Fig. 4. Experimental tracking results of link 2.

TABLE I

EXPERIMENTAL PERFORMANCE COMPARISON OF MIFNAC WITH OTHER CONTROLLERS

Method	# of Rules /Neurons	IAE					
		$t \in [0, t_5]$	$[0, t_1]$	$[t_1, t_2]$	$[t_2, t_3]$	$[t_3, t_4]$	$[t_4, t_5]$
PID Controller	-	4.8169	1.0252	0.9781	0.9093	0.9448	0.9595
Adaptive Controller [11]	-	3.2209	1.5100	0.4481	0.6071	0.3667	0.2891
NAC [7]	121	3.2050	0.6506	0.3890	0.6335	0.7446	0.7572
MNAC [7]	121	2.0778	0.6653	0.3972	0.2611	0.3943	0.3598
IFNAC	4–11	4.0770	0.9104	1.5901	0.2983	0.5317	0.7465
MIFNAC	3–9	1.7686	0.6348	0.4250	0.1853	0.3263	0.1972

scheme achieves better tracking performance with less fuzzy rules.

REFERENCES

[1] F. L. Lewis, A. Yeşildirek, and K. Liu, “Multilayer neural-net robot controller with guaranteed tracking performance,” *IEEE Trans. Neural Netw.*, vol. 7, no. 2, pp. 388–399, Mar. 1996.

[2] E. Kim, “Output feedback tracking control of robot manipulators with model uncertainty via adaptive fuzzy logic,” *IEEE Trans. Fuzzy Syst.*, vol. 12, no. 3, pp. 368–378, Jun. 2004.

[3] S. Wu, M. J. Er, and Y. Gao, “A fast approach for automatic generation of fuzzy rules by generalized dynamic fuzzy neural networks,” *IEEE Trans. Fuzzy Syst.*, vol. 9, no. 4, pp. 578–594, Aug. 2001.

[4] C.-S. Chen, “Dynamic structure neural-fuzzy networks for robust adaptive control of robot manipulators,” *IEEE Trans. Ind. Electron.*, vol. 55, no. 9, pp. 3402–3414, Sep. 2008.

[5] K. S. Narendra, O. A. Driollet, M. Feiler, and K. George, “Adaptive control using multiple models, switching and tuning,” *Int. J. Adaptive Control and Signal Process.*, vol. 17, no. 2, pp. 87–102, Mar. 2003.

[6] C.-Y. Lee and J.-J. Lee, “Adaptive control for uncertain nonlinear systems based on multiple neural networks,” *IEEE Trans. Syst., Man, Cybern. B*, vol. 34, no. 1, pp. 325–333, Feb. 2004.

[7] ——, “Multiple neuro-adaptive control of robot manipulators using visual cues,” *IEEE Trans. Ind. Electron.*, vol. 52, no. 1, pp. 320–326, Feb. 2005.

[8] C.-H. Kim, M.-S. Kim, and J.-J. Lee, “Incremental hyperplane-based fuzzy clustering for system modeling,” in *Proc. IEEE Conf. Ind. Electron. Soc.*, Taipei, Taiwan, Nov. 2007, pp. 614–619.

[9] H. Ying, “General SISO Takagi-Sugeno fuzzy systems with linear rule consequent are universal approximators,” *IEEE Trans. Fuzzy Syst.*, vol. 6, no. 4, pp. 582–587, Nov. 1998.

[10] N. Sadegh and R. Horowitz, “Stability and robustness analysis of a class of adaptive controllers for robotic manipulators,” *Int. J. Robot. Res.*, vol. 9, no. 3, pp. 74–92, Jun. 1990.

[11] J.-J. E. Slotine and W. Li, “Adaptive manipulator control: A case study,” *IEEE Trans. Autom. Control*, vol. 33, no. 11, pp. 995–1003, Mar. 1988.