

Nonlinear Control of Robot Manipulators Using Adaptive Fuzzy Sliding Mode Control

Feng-Yih Hsu¹ and Li-Chen Fu^{1,2}

Dept. of Electrical Engineering¹
Dept. of Computer Science & Information Engineering²
National Taiwan University, Taipei, Taiwan, R.O.C.

Abstract

This paper presents an adaptive robust fuzzy control architecture for robot manipulators. The control objective is to adaptively compensate for the unknown nonlinearity of robot manipulators, which is represented as a fuzzy rule-base consisting of a collection of if-then rules. The algorithm embedded in the proposed architecture can automatically update fuzzy rules and, consequently, it is guaranteed to be globally stable and to drive the tracking errors to a neighborhood of zero. Focused on realization, hardware limitations such as traditional long computation time and excessive memory-space usage are also relaxed by incorporating heuristic concepts, which reveals the flexible feature of this architecture. The present work is applied to the control of a five degree-of-freedom (DOF) articulated robot manipulator. Experiment results show that the proposed control architecture is featured in fast convergence.

1 Introduction

During the past decade, intelligent control methodologies have gradually been recommended to solve a number of complicated problems, in particular, to the problem of controlling robot manipulators which are hard by conventional control methodologies to handle or at the price of complex implementation. Those methodologies often use biologically motivated techniques and processes, and are referred to as neural networks, or some learning schemes [6]-[9]. Unlike general conventional schemes based on a complete theory and algorithmic structure, they are in general hardly evaluated. Therefore, it is imperative to make efforts on bridging the gap between the conventional control schemes and the intelligent ones [1]. Recently, analysis based intelligent control has attracted enormous research interests [2]-[4]. Ideas behind those schemes are to strengthen their theoretic basis but at the price of expensive implementation by using massive networks and extensive rule-tables or complex functions, which lead to difficulties in real implementation due to hardware limitations such as long computation time and excessive memory-space usage. On the other hand, the heuristic nature of intelligent control often has much benefits for a controlled system. Hence, an integrated consideration is suggested in this paper.

In this paper, we present a new fuzzy control architecture for the control of a robot manipulator. It is known that fuzzy logic controllers (FLC) have been widely applied in industry, for instance, applying fuzzy PD controller to robot manipulators. An important

advantage of using FLC is that fuzzy theories can capture the approximate, qualitative aspects of human knowledge and reasoning. Apparently, such control provides a rather feasible alternative for controlling a plant like a robot manipulator which is an extremely complex mechanical system. On the other hand, adaptive control and variable structure control, separately or both together, had been successfully applied to robot manipulators with appealing performance [10],[11]. Hence, an adaptive fuzzy sliding mode control (AFSMC) scheme for robot manipulators is proposed in this paper, where the variable structure control is to ensure that the overall system is stable whereas the adaptive control is to recover the control performance in face of the parametric uncertainty. Furthermore, the nonlinear sliding variables consisting of fuzzy PD rules will replace the conventional linear sliding variables because the former is more flexible than the conventional latter. Hence, the control synthesis proposed here incorporates the nonlinear fuzzy sliding surface approach into the present AFSMC.

This paper is organized as follows: Section 2 formulates the general control problem for robot manipulators. In section 3, the control algorithm is given under some assumptions and stability is analyzed. Section 4 shows experiment results on controlling a five DOF robot manipulator. Finally, some concluding remarks are made in section 5.

2 Problem Formulation

Consider a class of articulated robot arms with n links, whose dynamic model is described by

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) + f(q, \dot{q}) = \tau \quad (1)$$

where q is the $n \times 1$ vector of the link relative displacements, and τ is the $n \times 1$ vector of torques applied to joints, $M(q)$ is the inertial matrix, $C(q, \dot{q})\dot{q}$ is the vector representing the Coriolis, centrifugal torques and $G(q)$ is the vector of gravity torques, $f(q, \dot{q})$ is the vector of joint friction torques. For emphasis on the flexibility of applying robot manipulators, the manipulator payload may not be necessarily known in advance and a variety of tasks need to be handled. Besides, the friction in the process of robot motion is hardly modeled precisely. This results in uncertainties of modeling robot manipulators. Generally, uncertainties are often denoted as a deviation between the nominal plant and the actual plant and are expressed

as follows: $\widetilde{M} = M - \widehat{M}$, $\widetilde{C} = C - \widehat{C}$, $\widetilde{G} = G - \widehat{G}$, $\widetilde{f} = f - \widehat{f}$, where \widehat{M} , \widehat{C} , \widehat{G} and \widehat{f} are defined as the functions of the nominal plant. Our aim is to track a desired trajectory, i.e., to force the joint vector $q(t) = [q_1, q_2, \dots, q_n]^T$ to follow a specified desired trajectory $q_d(t) = [q_{1d}, q_{2d}, \dots, q_{nd}]^T(t)$. Hence, the tracking error vector $e = q_d - q = [e_1, e_2, \dots, e_n]^T$ is defined. At the beginning, a nominal compensator $\widehat{\tau}$ based on the computed torque methodology is designed as:

$$\widehat{\tau} = \tau_{pd} + \widehat{C}(q, \dot{q})\dot{q}_r + \widehat{G}(q) + \widehat{f}(q, \dot{q}) + \widehat{M}(q)\dot{q}_r, \quad (2)$$

where $\tau_{pd} = Ks$ is defined as a PD compensator, $s = [s_1, \dots, s_n]^T = \dot{e} + \lambda e$ is defined as a sliding mode vector and λ is a diagonal positive matrix with its diagonal elements denoted as $\{\lambda_1, \lambda_2, \dots, \lambda_n\}$, and K is a diagonal positive matrix, $q_r = \dot{q}_d + \lambda e$ is an additional available vector. Further, let the real controller $\tau = \widehat{\tau} + \widetilde{\tau}$. Then, a dynamic equation with sliding modes is given

$$M\dot{s} = -Ks - Cs + h(q, \dot{q}) + \widetilde{M}\dot{q}_r + \widetilde{C}s - \widetilde{\tau}, \quad (3)$$

where $h(q, \dot{q}) = \widetilde{C}\dot{q} + \widetilde{G} + \widetilde{f}$ is denoted as an unknown nonlinear vector. Hence, in addition to the nominal compensator mentioned above, it still needs an extra compensator to compensate for uncertainties of a robot manipulator. From a practical standpoint, h will be approximated by \widehat{h} as closely as possible. Hence, general nonlinear compensation function with sliding mode is given as follows:

$$\widetilde{\tau} = \tau_h + \tau_s, \quad (4)$$

where τ_h is a $n \times 1$ vector of compensating h , τ_s is a $n \times 1$ vector of robustifying remained uncertainties. Let index k represent the k -th element of a vector, then the k -th elements of τ_h and τ_s are given as $\tau_{hk} = \widehat{h}_k$ and $\tau_{sk} = (U_{sk}(q, \dot{q}) + U_M(q) \|\dot{q}_r\|_\infty) \text{sgn}(s_k)$, where $U_s \in \mathbb{R}^{n \times 1}$ and $U_M \in \mathbb{R}^1$ satisfy $U_{sk} \geq |\epsilon_s|$ and $U_M \geq \epsilon_M$, and $\epsilon_s = \epsilon_h + \widetilde{C}s$, $\epsilon_h = h - \tau_h$ and $\epsilon_M(q) = \|\widetilde{M}(q)\|_\infty$, then the tracking errors will exponentially converge to zero. Since $h(q, \dot{q})$ is hard to be approximated by using conventional control schemes, some intelligent control concepts need to be adopted. Moreover, since τ_s is not continuous, undesirable chattering may appear due to excessive magnitude design of τ_s . Here a suitable τ_s is required not only ensuring the system robustness but also the performance of the manipulator system.

On the other hand, λ is usually a constant matrix and its choice is often subjected to the hardware environment or software implementation such as sampling time of the control servo. Hence, the value of λ is often determined by the results of the simulation or experiment. From a practical standpoint, a nonlinear sliding variable which consists of a collection of fuzzy rules can more easily satisfy various consideration of the actual system. This result is illustrated by the fact that the architecture of the fuzzy PD controller is indeed more flexible than the crisp PD controller. Therefore, in this paper, an AFSMC is proposed to perform exactly as such an intelligent nonlinear controller.

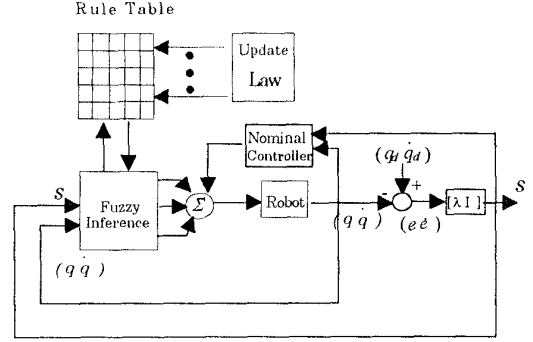


Figure 1: The architecture of AFSMC

3 AFSMC for Robot Manipulators

In this section, the AFSMC for a robot manipulator is analyzed. A fuzzy knowledge representation of the uncertainties of a robot manipulator is first described, which constitutes the main architecture of AFSMC. Next, a nonlinear sliding mode vector which consists of a collection of fuzzy PD rules is designed. Finally, AFSMC is analyzed by using adaptive robust control theory.

Referring to section 2, a robust fuzzy control law is given as follows:

$$\tau(t) = \widehat{\tau} + \tau_h^* + \tau_s^*, \quad (5)$$

where $\widehat{\tau} = Ks + \widehat{C}q_r + \widehat{G} + \widehat{f} + \widehat{M}\dot{q}_r$, $\tau_h^* = \widehat{h}(q, \dot{q})$, which is an optimal approximation of $h(q, \dot{q})$ in terms of the fuzzy knowledge representation, the k -th element of τ_s^* is given as $\tau_{sk}^* = (|\epsilon_s| + \epsilon_M \|\dot{q}_r\|_\infty) \text{sgn}(s_k)$, and τ_h , τ_s in an AFSMC are designed to approximate τ_h^* , τ_s^* , respectively, as shown in Fig. 1.

3.1 Fuzzy Knowledge Representation for Uncertainties

In this subsection, the main architecture of AFSMC for a robot manipulator is described. As a general description of fuzzy knowledge representation [4], a fuzzy rule-base consists of a collection of fuzzy *If-then* rules. Referring to section 2, the control objective is to compensate for the unknown nonlinear vector $h(q, \dot{q})$. First let $u = [q^T \dot{q}^T]^T = [u_1, \dots, u_{2n}]^T$ be denoted as an input linguistic vector in the discourse universe U_u and $L_j = \{L_j^1, L_j^2, \dots, L_j^{\alpha_j}, \dots, L_j^{l_j}\}$ as a family of fuzzy sets associated with the membership functions $\mu_{L_j^{\alpha_j}}$ (see Fig. 2) with respect to the variable u_j , where $L_j^{\alpha_j}$ is a fuzzy set in L_j . Besides, the supports of the family of fuzzy sets, L_j , are denoted as $U_j = \{U_j^1, U_j^2, \dots, U_j^{\alpha_j}, \dots, U_j^{l_j}\}$, where $U_j^{\alpha_j}$ is a point support satisfying $U_j^1 < U_j^2 < \dots < U_j^{\alpha_j} < \dots < U_j^{l_j}$ (also see Fig. 2). Let L be defined as a product set of $L = \prod_{i=1}^n L_i$, consisting of the families of fuzzy sets,

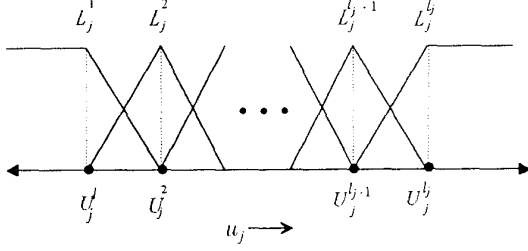


Figure 2: Fuzzy enviroment of the linguistic variable

L_i , $i = 1, \dots, 2n$, and U be defined as a product set of $U = \prod_{j=1} U_j$, consisting of the families of support points. Then, an example of the i -th fuzzy rule is represented as follows:

$$R[i]: \text{If } u \text{ is } L^{\alpha(i)}, \text{ then } w \text{ is } Q^{\beta(i)}, \quad (6)$$

where $L^{\alpha(i)} \in L$, $\alpha(i) = \alpha_{1(i)} \times \dots \times \alpha_{2n(i)}$ is a product index associated with the i -th rule, $w = \tau_h = [w_1, \dots, w_n]^T$ is denoted as an output linguistic vector, and $Q^{\beta(i)} \in Q$, $\beta(i) = \beta_{1(i)} \times \dots \times \beta_{n(i)}$ is a product index associated with the i -th rule, Q is a product set of $Q = \prod_{k=1} Q_k$, consisting of the families of fuzzy sets, Q_k , $k = 1, \dots, n$, where $Q_k = \{Q_k^1, Q_k^2, \dots, Q_k^{\beta_k}, \dots, Q_k^{\tau_k}\}$ is denoted as a family of fuzzy sets associated with the membership functions $\mu_{Q_k^{\beta_k}}$ with respect to the output variable w_k with $Q_k^{\beta_k}$ being a fuzzy set in the family Q_k . And, let the supports of the family of fuzzy sets, Q_k , be denoted as $W_k = \{W_k^1, W_k^2, \dots, W_k^{\beta_k}, \dots, W_k^{\tau_k}\}$, where $W_k^{\beta_k}$ is a point support satisfying $W_k^1 < W_k^2 < \dots < W_k^{\beta_k} < \dots < W_k^{\tau_k}$. Furthermore, the i -th rule is fired with a weighting function $\rho_i(u)$, which is determined by membership functions and a compositional operator. Hence, $\rho_i(u)$ can be expressed as follows:

$$\rho_i(u) = \begin{cases} \mu_{L_1^{\alpha_{1(i)}}}(u_1) \cdots \mu_{L_{2n}^{\alpha_{2n(i)}}}(u_{2n}), & \text{if sup-product operator;} \\ \min\{\mu_{L_1^{\alpha_{1(i)}}}(u_1), \dots, \mu_{L_{2n}^{\alpha_{2n(i)}}}(u_{2n})\}, & \text{if sup-min operator.} \end{cases} \quad (7)$$

where $\mu_{L_j^{\alpha_{j(i)}}}(u_j)$ is denoted as a membership function, which is a positive function with $\mu_{L_j^{\alpha_{j(i)}}}(u_j) \leq 1$ and reaches the maximum value when $u_j = U_j^{\alpha_{j(i)}}$ as shown in Fig. 2. In expression (7), the compositional operator is selected to be either the sup-product or the sup-min operator. Finally, using defuzzification function, the output variable w_k is expressed as follows:

$$w_k = \frac{\sum_{i=1}^{\Upsilon} D_{k_i}(\rho_i(u)) W_k^{\beta_{k(i)}}}{\sum_{i=1}^{\Upsilon} D_{k_i}(\rho_i(u))}, \quad (8)$$

where $D_{k_i}(\rho_i(u))$ is denoted as a defuzzification function with the point support $W_k^{\beta_{k(i)}}$ for the i -th rule commonly expressed as follows:

$$D_{k_i}(\rho_i(u)) = \begin{cases} \rho_i(u), & \text{if center-average-defuzzifier;} \\ \int_{-\infty}^{\infty} \mu_{Q_k^{\beta_{k(i)}}}(w_k) dw_k - \\ \int_{w_k^{-*}}^{w_k^{*}} [\mu_{Q_k^{\beta_{k(i)}}}(w_k) - \rho_i(u)] dw_k, & \\ \text{if center-of-area defuzzifier;} \end{cases} \quad (9)$$

where w_k^{-*} and w_k^{*} are solution values of w_k satisfying $\rho_i(u) = \mu_{Q_k^{\beta_{k(i)}}}(w_k)$ with $w_k^{*} \geq w_k^{-*}$ given the values u and Υ is the total number of rules generally equal to $l_1 \times l_2 \times \dots \times l_{2n}$. Then, we denote a fuzzy basis function $\xi_{k_i}(u)$ expressed as follows:

$$\xi_{k_i}(u) = \frac{D_{k_i}(\rho_i(u))}{\sum_{i=1}^{\Upsilon} D_{k_i}(\rho_i(u))} \quad (10)$$

so that equation (8) can be rewritten as:

$$w_k = \sum_{i=1}^{\Upsilon} W_k^{\beta_{k(i)}} \xi_{k_i}(u) = \Theta_k^T \xi_k(u), \quad (11)$$

where $\Theta_k = [W_k^{\beta_{k(1)}}, W_k^{\beta_{k(2)}}, \dots, W_k^{\beta_{k(\Upsilon)}}]^T$ is regarded as a parameter vector and $\xi_k = [\xi_{k_1}, \xi_{k_2}, \dots, \xi_{k_{\Upsilon}}]^T$ is regarded as a regressor vector. For simple notations, we define a product operator \otimes for two matrices $A \in \mathbb{R}^{n \times p}$, $B \in \mathbb{R}^{p \times n}$ as

$$A \otimes B = [A_1^T B_1, \dots, A_k^T B_k, \dots, A_n^T B_n]^T \quad (12)$$

where A_k is the k -th row vector of A and B_k is the k -th column vector of B , respectively. Hence, we denote the matrices $\Theta \in \mathbb{R}^{n \times \Upsilon}$, $\xi \in \mathbb{R}^{\Upsilon \times n}$ with the k -th row vector denoted as Θ_k and the k -th column vector denoted as ξ_k , respectively. Then the output vector w is expressed as

$$w = \Theta \otimes \xi. \quad (13)$$

Apparently, when Υ is large, the computation time and memory-space usage must be considered in real implementation. As a result, we focus our attentions on computation time first, then the number of the fired rules and that of the total rules should be clearly distinguished.

First, define the domain set of the total rules as $E = \{u : U_j^1 \leq u_j \leq U_j^{l_j}, j = 1, \dots, 2n\}$ and collect the indices of fired rules as a set of integer indices $I = \{i : \rho_i(u) > 0\}$. If $u \in E$, fuzzy rules are designed to compensate the unknown functions. Partition this domain into finite $2n$ - cells, which is defined as

$$E_{\alpha} = \{u : U_j^{\alpha_j} \leq u_j \leq U_j^{\alpha_j+1}, j = 1, \dots, 2n\},$$

where $\alpha = \alpha_1 \times \dots \times \alpha_j \times \dots \times \alpha_{2n}$ is a product index and satisfy $U^{\alpha} \in U$, $U^{\alpha_j} < U_j^{l_j}$. To union the all collection of E_{α} , we can obtain $E = \bigcup_{U^{\alpha} \in U} E_{\alpha}$. A δ - box is defined as

$$\Omega_{\alpha}(U^{\alpha}, \delta(u)) = \{u : U_j^{\alpha_j} \leq u_j \leq U_j^{\alpha_j} + \delta_j(u)\},$$

where $\delta(u) \in \mathbb{R}^{2n \times 1}$, the j -th element of $\delta(u)$ is defined as $\delta_j(u) = U_j^{\alpha_j+1} - U_j^{\alpha_j}$ and $j = 1, \dots, 2n$. Define a set of corner points of δ box Ω_α , $P_\alpha = \{U^{c(u)} : c(u) \in \prod_j^{2n} \{\alpha_j, \alpha_j + 1\}\}$. Hence the number of P_α , $\#n(P_\alpha)$ is equal to 2^{2n} .

Consider the membership function (Fig. 2):

$$\mu_{L_j^{\alpha_j}}(u_j) = \begin{cases} 1, & \text{as } u_j = U_j^{\alpha_j}; \\ 0, & \text{as } u_j \geq U_j^{\alpha_j+1} \text{ or } u_j \leq U_j^{\alpha_j-1}; \\ x_j^{\alpha_j}, & x_j^{\alpha_j} \in (0, 1), \text{ otherwise,} \end{cases} \quad (14)$$

for $\alpha_j = 1, 2, \dots, l_j$ and $j = 1, 2, \dots, 2n$, then the following proposition is satisfied:

Proposition 1 *If the membership functions are given as equation (14) and u is the linguistic vector and $u \in E$ then exists a δ -box, $\Omega_\alpha(U^\alpha, \delta(u))$ such that $u \in \Omega_\alpha(U^\alpha, \delta(u))$, and the number of fired fuzzy rules $\#n(I)$ will be smaller than or equal to the number of corner points of this δ -box, $\#n(P_\alpha)$.*

Based on Proposition 1., equation (10) can be replaced by:

$$\xi_{k_i}(u) = \begin{cases} \frac{D_{k_i}(\rho_i(u))}{\sum_{i \in I} D_{k_i}(\rho_i(u))}, & u \in \Omega_\alpha; \\ 0, & \text{otherwise,} \end{cases} \quad (15)$$

Hence, computation time is drastically reduced, especially, as Υ is very large.

3.2 Fuzzy Sliding Mode Control for Robot Manipulators

In this subsection, we will consider the design of PD compensator τ_{pd} by fuzzy knowledge representation. A linear PD compensator always plays an important role in the stabilization and noise rejection of the mechanical systems. But it is hard to properly tune the PD gain to overcome the varying dynamic process of robot manipulators for various tasks, and even if a proper gain tuning is achieved, it generally leads to the high gain condition. Therefore, a suitable gain scheduling based on the expert experience is more applicable to the actual system than the fixed gain. Besides, nonlinear sliding mode variables based on this gain scheduling can be more easily derived. This fact makes the representation of the sliding mode variables become flexible and easily satisfy the practical requirement for robot manipulators system.

Let $a = [e^T \dot{e}^T]^T$ be an input linguistic vector and the PD gains, $k_p = [k_{p1}, \dots, k_{pn}]^T$ and $k_d = [k_{d1}, \dots, k_{dn}]^T$, be two output linguistic vectors. Then, an example of fuzzy PD controller is expressed as follows:

$$R_{PD}[i]: \text{ If } a \text{ is } A^{\alpha(i)}, \text{ then } k_p \text{ is } P^{\beta(i)} \text{ and } k_d \text{ is } D^{\gamma(i)}, \quad (16)$$

where A , P , and D are product sets of $A = \prod_{j=1}^{2n} A_j$, $P = \prod_{k=1}^{2n} P_k$, $D = \prod_{k=1}^{2n} D_k$, consisting of the families of fuzzy sets, A_j , $j = 1, \dots, 2n$, P_k , D_k , $k = 1, \dots, n$,

for $A_j = \{A_j^1, \dots, A_j^{l_{aj}}\}$, $P_k = \{P_k^1, \dots, P_k^{r_{pk}}\}$, $D_k = \{D_k^1, \dots, D_k^{r_{dk}}\}$. Let K_{pk} , K_{dk} be denoted as the supports of the family of fuzzy sets, P_k , D_k , respectively. From subsection 3.1, we can obtain

$$k_{pk} = \sum_{i=1}^{\Upsilon_{pd}} K_{pk}^{\beta_{k(i)}} \xi_{pk_i}(a) \quad (17)$$

$$k_{dk} = \sum_{i=1}^{\Upsilon_{pd}} K_{dk}^{\gamma_{k(i)}} \xi_{dk_i}(a). \quad (18)$$

where ξ_{pk_i} , ξ_{dk_i} are the fuzzy basis functions associated with the i -th rule, R_{pd} is the total rule number. Let $K_{pk}^{r_{pk}} > \dots > K_{pk}^1 > \sigma_p > 0$, $\sigma_d > K_{dk}^{r_{dk}} > \dots > K_{dk}^1 > 0$, where σ_p, σ_d are positive constants. Then, the k -th control input variable $\tau_{pdk} = k_{pk}e_k + k_{dk}\dot{e}_k$ is given. Let $s_k = \dot{e}_k + \lambda_k(e, \dot{e})e_k$, where $\lambda_k(e, \dot{e}) = \frac{k_{pk}}{k_{dk}} > \frac{\sigma_p}{\sigma_d} > 0$ is given. This implies that s_k is a stable sliding variable. Finally, the k -th control input variable is rewritten as $\tau_{pdk} = k_{pk}s_k$.

3.3 Adaptive Fuzzy Sliding Control for Robot Manipulators

In this subsection, the fuzzy theory is combined with adaptive robust control architecture. In contrast to general conventional control schemes, we use fuzzy representation to approximate the unknown functions in section 2. Based on mild assumptions, adaptive fuzzy sliding mode control (AFSMC) is not only proved to be globally stable but also exhibits the nature of intelligent control.

Let $v = [q_c^T \dot{q}_c^T s^T]^T = [v_1, \dots, v_k, \dots, v_{3n}]^T$ be denoted as another input linguistic vector with respect to τ_s in the discourse universe U_v , and the supports of the family of fuzzy sets with respect to v_k are denoted as

$$V_k = \{V_k^1, V_k^2, \dots, V_k^{m_k}\}, \quad \text{where} \\ V_k^1 < V_k^2 < \dots < V_k^{m_k} \quad \text{and} \quad k = 1, 2, \dots, 3n.$$

Let τ_s is denoted as $\tau_s = [\tau_{s1}, \dots, \tau_{sk}, \dots, \tau_{sn}]^T$, where τ_{sk} be expressed as follows:

$$\begin{aligned} \tau_{sk} &= \sum_{i=1}^{\Upsilon_s} \Theta_{sk_i} \xi_{sk_i}(v) + \Theta_{Mk} \xi_{Mk}(v) \|\dot{q}_r\|_\infty \\ &= \Theta_{sk}^T \xi_{sk} + \Theta_{Mk}^T \xi_{Mk} \|\dot{q}_r\|_\infty, \end{aligned} \quad (19)$$

where Θ_{sk} and Θ_{Mk} are denoted as parameter vectors, ξ_{sk} , ξ_{Mk} are denoted as regressor vectors as equation (10), and Υ_s is denoted as the total number of rules for τ_s . Hence, the robust compensator vector τ_s can be expressed as follows:

$$\tau_s = \Theta_s \otimes \xi_s + \Theta_M \otimes \xi_M \|\dot{q}_r\|_\infty, \quad (20)$$

where the matrices Θ_s, Θ_M , $\in \mathbb{R}^{n \times \Upsilon_s}$, consist of the k -th row vectors Θ_{sk}, Θ_{Mk} , respectively, and ξ_s, ξ_M ,

$\in \mathbb{R}^{Y_s \times n}$, consist of the k -th column vector ξ_{sk} , ξ_{Mk} for $k = 1, \dots, n$, respectively. Our goal is to design optimal parameter matrices such that the controlled system has minimal tracking errors and robust features. At the beginning, we must prove that bounds on approximation errors depend on our design. The k -th row vectors of optimal parameter matrices are defined as follows:

$$\Theta_k^* = \arg \min[\sup_{z \in E} |\Theta_k^T \xi_k - h_k|] \quad (21)$$

$$\begin{aligned} \Theta_{sk}^* &= \arg \min[\sup_{v \in E_s} \Theta_{sk}^T \xi_{sk} \operatorname{sgn}(s_k) \\ &\geq |\epsilon_{sk}|] \end{aligned} \quad (22)$$

$$\begin{aligned} \Theta_{Mk}^* &= \arg \min[\sup_{v \in E_s} \Theta_{Mk}^T \xi_{Mk} \operatorname{sgn}(s_k) \\ &\geq \epsilon_M] \end{aligned} \quad (23)$$

where E , E_s are compact domain sets of input linguistic vectors u , v in all rules and are expressed as follows:

$$E_s = \{v : V_k^1 \leq v_k \leq V_k^{m_k}, k = 1, 2, \dots, 3n\}$$

Just like some other general approximation function, solution to the optimization problems expressed in (21) will always exist, but solutions to the problems in (22), (23) may not exist in the neighborhood of $s = 0$. Hence, we purposely specify one of the support points with respect to the k -th sliding mode variable s_k as $V_{2n+k}^{\alpha_{2n+k}} = 0$ so that the elements of Θ_{sk}^* will be classified into two groups: one is $s_k > 0$ and the other $s_k < 0$. This will make the problem formulation (22), (23) more reasonable and guarantee the existence of their solutions. Later, we will replace the definition, $V_{2n+k}^{\alpha_{2n+k}} = 0$ with a more relaxed one by defining a deadzone of s_k instead. To simplify the problem, mild assumptions are given as follows:

Assumptions:

- $h \in C^1$ (continuously differentiable)
- $\|h\|_\infty < h^U(u)$
- $\|\tilde{M}\|_\infty < M^U(u)$
- $\|\tilde{C}\|_\infty < C_s^U(u)$

Based on those assumptions, Proposition 2 is given.

Proposition 2 *If the linguistic vector u falls into δ -box, $\Omega_\alpha(U^\alpha, \delta(u))$, then the k -th element of optimal approximation error vector $\epsilon_h(u)$ will be bounded by $|\epsilon_{hk}| \leq g_k^T(u)\delta(u)$, where $\epsilon_h(u) = h - \Theta^* \otimes \xi$, g_k is the k -th row vector of matrix $g(u)$, $g(u) \in \mathbb{R}^{n \times 2n}$, and its element is defined as $g_{ij} = \sup_{u \in \Omega_\alpha} [|\frac{\partial h_i(u)}{\partial u_j}|]$ for $i = 1, \dots, n$, $j = 1, \dots, 2n$.*

Furthermore, let the control law be redesigned as

$$\tau = d(t)(\tau_h + \tau_s) + (1 - d(t))U_s(z_s) + \hat{\tau}, \quad (24)$$

where

$$k_p(e, \dot{e}) > \sigma_p > 0 \quad \text{and} \quad d(t) = \begin{cases} 1, & \text{as } v \in E_s; \\ 0, & \text{otherwise.} \end{cases} \quad (25)$$

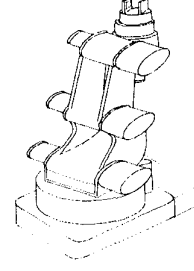


Figure 3: The diagram of robot manipulators

$$U_s(v) = (h^U + M^U \|\dot{q}_r\|_\infty + C_s^U \|s\|_\infty) \operatorname{sgn}(s) \quad (26)$$

Update laws are given as follows :

$$\dot{\Theta} = r \operatorname{diag}(s_\Delta) \xi(u)^T \quad \text{as } u \in E \quad (27)$$

$$\dot{\Theta}_s = r \operatorname{diag}(s_\Delta) \xi_s^T(v) \quad \text{as } v \in E_s \quad (28)$$

$$\dot{\Theta}_M = r \|\dot{q}_r\|_\infty \operatorname{diag}(s_\Delta) \xi_M^T(v) \quad \text{as } v \in E_s \quad (29)$$

for some $r > 0$ and

$$\begin{aligned} s_\Delta &= \begin{cases} s - \alpha, & \text{as } s < \alpha; \\ s - \beta, & \text{as } s > \beta; \\ 0, & \text{otherwise.} \end{cases} \quad (30) \\ s_\Delta &= \dot{s} \end{aligned}$$

where $\alpha, \beta \in \mathbb{R}^n$ are some constant vectors and their the k -th elements satisfying $\alpha_k \leq V_{2n+k}^{\alpha_{2n+k}} \leq 0 \leq V_{2n+k}^{\alpha_{2n+k}+1} \leq \beta_k$.

Theorem 1 *If the control law and the update law are given as in equation (24) and in equations (27)-(30), then the tracking errors will asymptotically converge to a neighborhood of zero.*

4 Experimental Results

A five degree-of-freedom (DOF) articulated robot arm is set up in the Intelligent Robot Laboratory of CS&IE in NTU as shown in Fig. 3. An AF-SMC is designed in assumption that inertial matrix M , Coriolis, centrifugal torques C , and gravity vector G of the arm are unknown. To show the effectiveness of adaptive robust law, τ_s , we will neglect the compensation function τ_h . The same desired trajectory $-22.5 + 22.5 \cos(\pi/2t)$ is given for five different joints. A simplified strategy for reducing memory-space usage is to decentralize the previous architecture and ignore \dot{q} , i.e. $\tau_{s_i}(z_s) = \tau_s(q_i, s_i)$. The triangular form and sup-min operator are selected as membership functions and compositional operators. The total rule number of AF-SMC is $5 \times r_q \times r_s = 5 \times 17 \times 17$, the sampling time of control servo is $1ms$, and results are listed in Fig. 4. At the beginning, we only use PD controller to compensate for the uncertainties in the first period of sine wave, and then incorporate the AF-SMC right after that the tracking errors are quickly driven toward zero. This shows that AF-SMC has a fast converging feature.

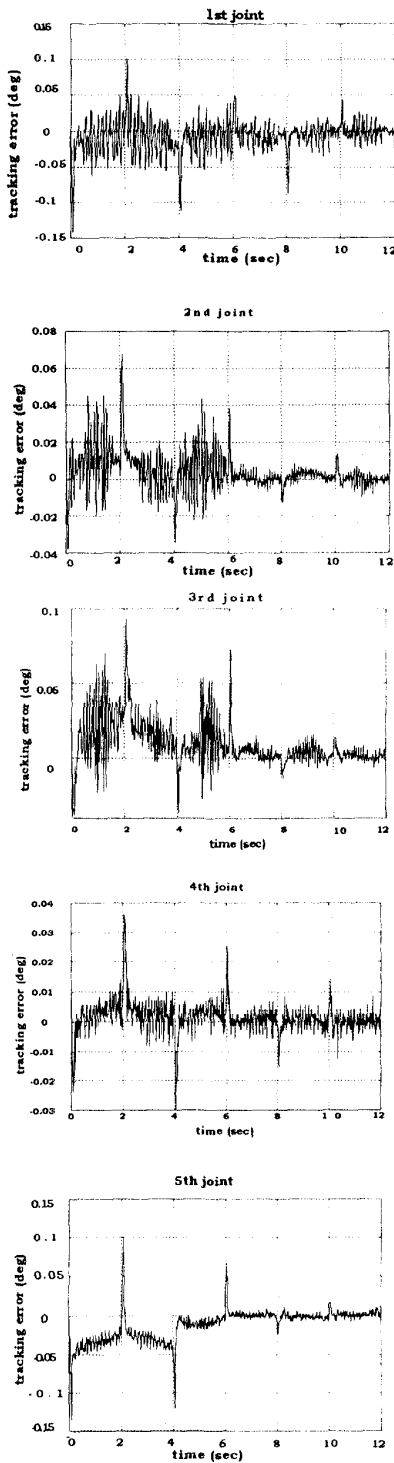


Figure 4: The experiment result: tracking error (deg) V.S. time (sec)

5 Conclusions

We had proposed a novel fuzzy controller design method, which can update fuzzy rules to compensate unknown uncertainty of robot manipulator and guarantee the stability of the controlled robot manipulator systems. Besides, a dexterous use of fuzzy mathematics made the complicated scheme implemented with computing efficiency. The future work is to apply this control scheme to the hybrid force and position control.

References

- [1] K. M. Passino, "Bridging the gap between Conventional and Intelligent Control", *IEEE Cont. Syst. Magaz.*, vol. 13, no. 3, pp. 12-18, 1993
- [2] R. M. Sanner and J-J E. Slotine, "Gaussian Networks for Direct Adaptive Control", *IEEE, Trans. on Neural Networks*, vol. 3, no. 6, pp. 837-863, 1992.
- [3] A. U. Levin. and K. S. Narendra, "Control of Nonlinear Dynamical Systems Using Neural Networks: Controllability and Stabilization", *IEEE, Trans. on Neural Networks*, vol. 4, no. 2, pp. 192-205, 1993.
- [4] L. X. Wang, "Stable Adaptive Fuzzy Control of Nonlinear Systems", *IEEE, Conf. on Decision and Control*, 1992
- [5] J-J E. Slotine and W. Li, *Applied Nonlinear Control*, Englewood Cliffs, NJ: Prentice Hall, pp. 278-284, 1991.
- [6] Sheng Liu and H. Asada, "Teaching and Learning of Deburring Robots Using Neural Networks" *IEEE Conference on Robotics and Automation*, pp. 339-345, 1993
- [7] S. Arimoto, "Learning Control Theory for Robotic Motion", *International Journal of Adaptive Control and Signal Processing*, vol. 4, pp. 543-564, 1990.
- [8] S. Kawamura, F. Miyazaki and S. Arimoto, "Realization of robot motion based on a learning method", *IEEE, Trans. on Syst., Man, Cybern.*, vol. 18, pp. 126-134, 1988.
- [9] R. Horowitz, Perry Li, "Multitask Robot Learning Control" *American Control Conference*, pp.2623-2628 1992
- [10] Nader Sadeh and R. Horowitz, "Stability and Robustness Analysis of a Class of Adaptive Controllers for Robotic Manipulators" *The International Journal of Robotics Research*, vol. 9, no. 3, pp. 74-92, 1990.
- [11] G. Niemeyer, J-J E. Slotine, "Performance in Adaptive Manipulator Control", *The International Journal of Robotics Research*, vol. 10, no. 2, pp. 149-161, 1991.
- [12] S. Sastry and M. Bodson, *Adaptive Control: Stability, Convergence, and Robustness*, Prentice Hall, pp.19 1989.