Adaptive Robust Fuzzy Control for Robot Manipulators

Feng-Yih Hsu[1] and Li-Chen Fu[1,2]

Dept. of Electrical Engineering[1]
Dept. of Computer Science & Information Engineering[2]
National Taiwan University, Taipei, Taiwan, R.O.C.

Abstract

This paper presents an adaptive robust fuzzy control architecture for robot manipulators motion. 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. Simulation 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 control problem of robot manipulators which conventional control methodologies are hard 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. 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 a plant like a robot manipulator which is extremely complex. On the other hand, adaptive control schemes had been successfully applied to robot manipulators with good performance [10],[11]. Hence, an adaptive fuzzy control scheme for robot manipulators is proposed in this paper. Furthermore, robust control concepts have to be adopted to ensure that the controlled system is stable. Thus, a synthesis is proposed by combining the adaptive fuzzy control approach with a robust control approach to constitute the final adaptive robust fuzzy control (ARFC).

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 the simulation results on controlling a five DOF robot manipulator. Finally, some concluding remarks are made in section 5.

2 Problem Formulation

In this section, we consider a class of articulated robot arms wit n links, whose dynamic model is described by

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

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. To 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: $M = \widehat{M} + \Delta M$, $C = \widehat{C} + \Delta C$,

$G = \widehat{G} + \Delta G$, $f = \widehat{f} + \Delta f$. Our aim is to track a desired trajectory, i.e., to force the joint vector $q(t) = [q_1, q_2, \cdots, q_n]^T$ to follow a specified desired trajectory $q_d(t) = [q_{1d}, q_{2d}, \cdots, q_{nd}]^T(t)$. Hence, the tracking error vector $e = q_d - q = [e_1, e_2, \cdots, e_n]^T$ is defined. At the beginning, a nominal controller $\widehat{u}$ based on the computed torque methodology is designed as :

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

where $s = \dot{e} + \lambda e$ is defined as a sliding mode vector and λ is a diagonal positive matrix with its diagonal elements denoted as $\{\lambda_1, \lambda_2, \cdots, \lambda_n\}$, and K is a diagonal positive matrix, $q_r = \dot{q}_d + \lambda e$ is an aditional available vector. Further, let the real controller $u = \widehat{u} + \Delta u$. Then, a dynamic equation with sliding modes is given

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

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

$$\Delta\tau = \tau_h + \tau_M + \tau_s, \quad (4)$$

where τ_h, τ_M, τ_s are given as $\tau_h = \widehat{h}$, $\tau_M = \widehat{\Delta M}\dot{q}_r$ and $\tau_s = U(q,\dot{q},\dot{q}_r)sgn(s)$. Assume that $\left|\tau_h - \widehat{h}\right| \leq |\epsilon_h(q,\dot{q})|$, $\left|\tau_M - \widehat{\Delta M}\dot{q}_r\right| \leq |\epsilon_M(q,\dot{q},\dot{q}_r)|$ and $U(q,\dot{q},\dot{q}_r) \geq |\epsilon_h(q,\dot{q})|, + |\epsilon_M(q,\dot{q},\dot{q}_r)|$, then the tracking errors will asymtotically converge to zero. Since $\widehat{h}$ and $\widehat{\Delta M}$ are hard to be approximated by using conventional control schemes. Thus, some intelligent control concepts may be adopted. Moreover, since τ_s is not continuous, serious chattering will occur owing to excessive τ_s design. Hence, a suitable τ_s will ensure both robustness and performance of the controlled robot manipulators. In this paper, we will regard ARFC as this intelligent nonlinear controller.

3 ARFC for Robot Manipulators

In this section, the ARFC for a robot manipulator is analyzed. First, a fuzzy knowledge representation of the uncertainties of a robot manipulator is described in subsection 3.1, which provides the main architecture of ARFC. In subsection 3.1, ARFC is analyzed by using adaptive robust control theory. At last, an enhanced fuzzy rule-base is designed, and the whole algorithm is shown in subsection 3.3.

3.1 Fuzzy Knowledge Representation for Uncertainties

In this subsection, the main architecture of ARFC 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, control objective is to compensate for the unknown nonlinear vector $h(q,\dot{q})$ and $\Delta M\dot{q}_r$. First, we will consider a simpler case in which only $h(q,\dot{q})$ is approximated by fuzzy knowledge representation. Let q be denoted as a fuzzy input vector and $Q_j = \{Q_j^1, Q_j^2, \cdots, Q_j^{r_{q_j}}\}$ be denoted as a support of fuzzy set with respect to the fuzzy input variable q_j, where Q_j^k is an element of the support satisfying $Q_j^1 < Q_j^2 < \cdots < Q_j^k < \cdots < Q_j^{r_{q_j}}$ and r_{q_j} is the number of supporting points of the set Q_j, $j = 1, 2, \cdots, n$. Similarly, $\dot{q}$ is also denoted as another fuzzy input vector with supports of the fuzzy set $DQ = \{DQ_j^1, DQ_j^2, \cdots, \cdots, DQ_j^{r_{dq_j}}\}$, where r_{dq_j} is the number of supporting points of corresponding fuzzy set $j = 1, 2, \cdots, n$. Then, the ith rule is represented as follows:

$$R[i]: \quad \text{If} \quad q \quad \text{is} \quad Q^{(\alpha_i)} \quad \text{and } \dot{q} \quad \text{is} \quad DQ^{(\beta_i)}$$
$$\text{then} \quad y = Y^{(\gamma_i)} \quad (5)$$

where y is denoted as a fuzzy output vector, and $Y_j = \{Y_j^1, Y_j^2, \cdots, Y_n^{r_y}\}$ is denoted as the support of the fuzzy set with respect to the fuzzy output y_j. The index i represents the label of rules, hence $Q^{(\alpha_i)}$, $DQ^{(\beta_i)}$, $Y^{(\gamma_i)}$ represent points of supports of the relevant fuzzy sets on the ith rule. Furthermore, every rule is fired with a weighting function $\mu_i(q,\dot{q})$, which is determined by membership functions and compositional operators. More in detail for clarity expression, let $z = [q^T\dot{q}^T]^T$ and supports of the fuzzy set $Z = Q \times DQ = \prod_j Q_j \times \prod_j DQ_j$, then $\mu_i(z)$ can be expressed as follows :

$$\mu_i(z) = \begin{cases} \rho(z_1 \mid Z_1^{\alpha_1(i)}) \bullet \cdots \bullet \rho(z_{2n} \mid Z_{2n}^{\alpha_{2n}(i)}), \\ \qquad \text{if sup-product operator;} \\ min\{\rho(z_1 \mid Z_1^{\alpha_1(i)}), \cdots \rho(z_{2n} \mid Z_{2n}^{\alpha_{2n}(i)})\}, \\ \qquad \text{if sup-min operator.} \end{cases} \quad (6)$$

where the membership function $\rho(z_j \mid Z_j^{\alpha_j(i)})$ is defined as a positive monotone decreasing function as fuzzy variable z_j is apart from $Z_j^{\alpha_j(i)}$ and $\rho(z_j \mid Z_j^{\alpha_j(i)}) \leq 1$, where $j = 1, \cdots, 2n$. In expression(6), the composional operator is selected to be either sup-product or sup-min operator. Finally, the output y is expressed as follows:

$$y = \frac{\sum_{i=1}^{R_t} \mu_i(z) Y^{(i)}}{\sum_{i=1}^{R_t} \mu_i(z)}, \quad (7)$$

where R_t is the total number of rules and generally equals $r_{q_1} \times \cdots \times r_{q_n} \times r_{dq_1} \times \cdots \times r_{dq_n}$ We normalize $\mu_i(z)$ by summation of the total weighting functions, *i.e.*,

$$\xi_i(z) = \frac{\mu_i(z)}{\sum_{i=1}^{R_t} \mu_i(z)}, \quad (8)$$

where $\xi_i(z)$ is denoted as a fuzzy basis function so that equation(7) is rewritten as

$$y = \sum_{i=1}^{R_t} Y^{(i)}\xi_i(z) = \sum_{i=1}^{R_t} \Theta_i^T \xi_i(z) = \Theta^T \xi(z), \qquad (9)$$

where $\Theta = [\Theta_1^T, \Theta_2^T, \cdots, \Theta_{R_t}^T]^T$ is denoted as a parameter matrix with dimension $R_t \times n$ and $\Theta_i = Y^{(i)^T}$ is denoted as its raw vetcor, and $\xi = [\xi_1, \xi_1, \cdots, \xi_{R_t}]^T$ is denoted as a regressor vector.

When the number of points supports of fuzzy sets, namely, r_{q_j}, r_{dq_j}, and link dimensions n is large, the rule number will become very large. Hence, computation time and memory-space have to be considered in those cases. Focusing our attentions on computation time, the domain set of the fired rules and the domain set of total rules must be made clear. This leads that the control law has to be simplified by suitable assumptions.

To simplify the control law, we set $\xi_i(z^*) = 1$ for the i-th rule, and $z^* \in Z$. This results in the following membership function:

$$\rho(z_j \mid Z_j^{\alpha_j}) = \begin{cases} 1, & \text{as } z_j = Z_j^{\alpha_j}; \\ 0, & \text{as } z_j > Z_j^{\alpha_j+1} \text{ or } z_j < Z_j^{\alpha_j-1}, \end{cases} \qquad (10)$$

where $Z_j^{\alpha_j}$ is the element of Z_j, which implies that rules are fired when

$$\rho(z_j \mid Z_j^{\alpha_j}) > 0 \quad \text{if} \quad Z_j^{\alpha_j-1} < z_j < Z_j^{\alpha_j+1}. \qquad (11)$$

Then, defining a compact domain set of fuzzy input vector in fired rules is:

$$Z^C = \{z \mid Z_j^{c_j} \le z_j \le Z_j^{c_j+1} \; j = 1, 2, ..., 2n\}, \quad (12)$$

where

$$c_j = \begin{cases} \alpha_j - 1, & \text{if } Z_j^{\alpha_j-1} < z_j < Z_j^{\alpha_j}; \\ \alpha_j, & \text{if } Z_j^{\alpha_j} < z_j < Z_j^{\alpha_j+1}. \end{cases} \qquad (13)$$

Therefore, eqation(8)is replaced by :

$$\xi_i(z) = \begin{cases} \frac{\mu_i(z)}{\sum_{i\in C}\mu_i(z)}, & z \in Z^C ; \\ 0, & \text{otherwise}, \end{cases} \qquad (14)$$

where C is denoted as the set of fired rules so that the total number of fired rules is 2^{2n}.

Based on this result, the total number of fired rules is reduced from $r_{q_1} \times \cdots \times r_{q_n} \times r_{dq_1} \times \cdots \times r_{dq_n}$ to 2^{2n} and only depends on the dimension of the fuzzy input vector. For example, consider a robot manipulator with one link and numbers of support are given as $r_{q_1} = 10$, $r_{dq_1} = 10$, then this leads to total number of rules $R_t = r_{q_1} \times r_{dq_1} = 10 \times 10 = 100$, but only $2^2 = 4$ fuzzy rules are fired. Hence, computation time is extensively reduced, especially, as R_t is very large.

3.2 Adaptive Robust Fuzzy 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 robust fuzzy control(ARFC) is not only proved to be globally stable but also exhibits the nature of intelligent control.

Referring to section 2 and only considering much simpler case, *i.e.* $\Delta M = 0$, then the control law is given as follows

$$\tau(t) = \hat{\tau} + \tau_h + \tau_s. \qquad (15)$$

where K is a diagonal positive matrix, $\hat{\tau} = Ks + \hat{C}q_r + \hat{G} + \hat{f} + \widehat{M}\dot{q}_r$, $\tau_s = \Delta Cs + U(q, \dot{q})sgn(s)$, $\tau_h = \Delta C\dot{q} + \Delta G + \Delta f$, and an ARFC is designed to approximate τ_h, τ_s as shown in Figure 1. To analyze the control in subsection 3.1, at first, a fuzzy environment has to be built. Hence, let $z_h = [q^T \dot{q}^T]^T$ be denoted as a fuzzy input vector with respect to h, $z_s = [q^T \dot{q}^T s^T]^T$ be denoted as another fuzzy input vector with respect to τ_s. Corresponding to those fuzzy input vectors, supports of fuzzy sets are given as follows:

$$Z_{h_j} = \{Z_{h_j}^1, Z_{r_{h_j}}^2, \cdots, Z_{h_j}^{r_{hj}}\}, \quad \text{where}$$
$$Z_{h_j}^1 < Z_{h_j}^2 < \cdots < Z_{h_j}^{r_{hj}} \quad \text{and} \quad j = 1, 2, \cdots, 2n;$$
$$Z_{s_k} = \{Z_{s_k}^1, Z_{s_k}^2, \cdots, Z_{s_k}^{r_{sk}}\}, \quad \text{where}$$
$$Z_{s_k}^1 < Z_{s_k}^2 < \cdots < Z_{s_k}^{r_{sk}} \quad \text{and} \quad k = 1, 2, \cdots, 3n.$$

Let control law be expressed as follows:

$$\tau_h \;=\; \theta_h = \sum_{i=1}^{R_h} \Theta_{h_i}^T \xi_{h_i}(z_h) = \Theta_h^T \xi_h; \qquad (16)$$

$$\tau_s \;=\; \theta_s = \sum_{i=1}^{R_s} \Theta_{s_i}^T \xi_{s_i}(z_s) = \Theta_s^T \xi_s, \qquad (17)$$

where θ_h, θ_s, are denoted as fuzzy output vectors, Θ_h, Θ_s are denoted as parameter matrices, and ξ_h, ξ_s are denoted as regressor vectors as eq(9). R_h, R_s are denoted as the total number of rule for τ_h and τ_s. Our goal is to design optimal parameter matrices and regressor vectors such that the controlled system has minimal tracking errors and robust features. In this subsection, we will only consider parameter vectors. In next subsection, the whole case is discussed. At the beginning, we must prove that bounds on approximation errors depend on ARFC design.

Optimal parameter vectors are defined as follows:

$$\Theta_h^* = arg\ min[sup_{z_h \in Z_h^B} |\Theta_h^T \xi_h(z_h) - \tau_h|] \qquad (18)$$

$$\Theta_s^* \;=\; arg\ min[sup_{z_s \in Z_s^B} \Theta_s^T \xi_s(z_s)$$

$$
\begin{aligned}
& -(|\Delta Cs| + U)sgn(s)] \quad \text{if} \quad s > 0 \\
= \; & arg\ min[sup_{z_s \in Z_s^B} - \Theta_s^T \xi_s(z_s) \\
& +(|\Delta Cs| + U)sgn(s)] \quad \text{if} \quad s < 0 \qquad (19)
\end{aligned}
$$

where Z_h^B, Z_s^B are compact domain sets of fuzzy input vector in all rules and are expressed as follows:

$$
\begin{aligned}
Z_h^B &= \{z_h \mid z_{h_j}^{1_j} \le z_{h_j} \le z_{h_j}^{r_{h_j}}, j = 1, 2, \cdots, 2n\} \\
Z_s^B &= \{z_s \mid z_{s_k}^{1_k} \le z_{s_k} \le z_{s_k}^{r_{s_k}}, k = 1, 2, \cdots, 3n\}
\end{aligned}
$$

Based on the analysis of subsection 3.1, not all rules of ARFC are simultaneously fired. The fired rules are located on a specified domain set of fuzzy input vector. Those sets are denoted as follows:

$$
\begin{aligned}
Z_h^C &= \{z_h \mid Z_{h_j}^{c_{h_j}} \le z_{h_j} \le Z_{h_j}^{c_{h_j}+1}, j = 1, 2, \cdots, 2n\}, \\
Z_s^C &= \{z_s \mid Z_{s_k}^{c_{s_k}} \le z_{s_k} \le Z_{s_k}^{c_{s_k}+1}, k = 1, 2, \cdots, 3n\},
\end{aligned}
$$

where indices c_{h_j} and c_{s_k} are defined as in eq(13) and C_h and C_s are denoted as sets of fired rules.

To simplify the problem, mild assumptions are given as follows:

Assumptions:

- $\Delta Cs \in c^1$ (continuously differentiable)
- $h(z_h) \in c^1$ (continuously differentiable)
- $\mid h \mid < h^U(z_h)$

Based on those assumptions, Proposition 1 is given.

Proposition 1 *If $z_h \in Z_h^B$, then $|\epsilon_h(z_h)| \le c_h(z_h)\Delta_h(z_h)$, where $\epsilon_h(z_h) = h - \sum_{i \in C_h} \Theta_{h_i}^* \xi_{h_i}(z_h)$, $c_h(z_h)$ is an $n \times 2n$ matrix, and its element is defined as $c_{h_{ij}} = sup_{z_h \in Z_h^C}[\left|\frac{\partial h_i(z_h)}{\partial z_{h_j}}\right|]$,*

$$
\Delta_h(z_h) = [Z_h^{c_1+1} - Z_h^{c_1}, \cdots, Z_{h_{2n}}^{c_{2n}+1} - Z_{h_{2n}}^{c_{2n}}]^T.
$$

Proof: Defining Z_h^* as supports of fuzzy set in fired rules. Then, let

$$
{\Theta^\star}_h^T \xi_h(z_h) = h(z_h) \quad as \quad z_h \in Z_h^* \qquad (20)
$$

Since, $\sum_{i \in c_h} \xi_{h_i}(z_h) = 1$, it results in the following:

$$
\begin{aligned}
min(\{\Theta_{h_i}^* \ as\ i \in C_h\}) &\le \sum_{i \in C_h} {\Theta_{h_i}^*}^T \xi_i(z_h) \\
&\le max(\{\Theta_{h_i}^* \ as\ i \in C_h\})
\end{aligned}
$$

which implies

$$
\left| h(z_h) - \sum_{i \in C_h} {\Theta_{h_i}^*}^T \xi_i(z_h)) \right| \le c_h(z_h)\Delta_h(z_h)
$$

Hence,

$$
\begin{aligned}
|\epsilon_h(z_h)| &= \left| h - \sum_{i \in C_h} {\Theta_{h_i}^*}^T \xi_i(z_h) \right| \\
&\le \left| h - \sum_{i \in C_h} {\Theta_{h_i}^*}^T \xi_i(z_h) \right| \\
&\le c_h(z_h)\Delta_h(z_h)
\end{aligned}
$$

Based on Proposition 1, approximation errors will be reduced by adjusting Δ_h. Hence, robust controller with high gain is not necessary to ARFC. Let the control law renewedly be changed to the following:

$$
\tau = d(t)(\tau_h + \tau_s) + (1 - d(t))U_s(z_s) + \hat{\tau}, \qquad (21)
$$

where

$$
d(t) = \begin{cases} 1, & \text{as } z_s \in Z_s^B; \\ 0, & \text{otherwise.} \end{cases}, \qquad (22)
$$

$$
U_s(z_s) = h^U(z_h) + |\Delta C(z_h)s|. \qquad (23)
$$

Update laws are given as follows :

$$
\begin{aligned}
\dot{\Theta}_h &= r s_\Delta \xi_h^T(z_h) \ as\ z_h \in Z_h^B \qquad (24) \\
\dot{\Theta}_s &= r s_\Delta \xi_s^T(z_s) \ as\ z_s \in Z_s^B \qquad (25)
\end{aligned}
$$

where $r > 0$ and

$$
\begin{aligned}
s_\Delta &= \begin{cases} s - S^{(\alpha_0)}, & \text{as } s < S^{(\alpha_0)}; \\ s - S^{(\alpha_0+1)}, & \text{as } s > S^{(\alpha_0+1)}; \\ 0, & \text{othewise.} \end{cases} \\
\dot{s}_\Delta &= \dot{s}
\end{aligned}
$$

where $S^{(\alpha_0)} < 0 < S^{(\alpha_0+1)}$, $S^{(\alpha_0)}$ and $S^{(\alpha_0+1)}$ are subsets of S and α_0 is its label. Hence, a deadzone is expressed as $\phi = [S^{\alpha_0} \ S^{(\alpha_0+1)}]$.

Proposition 2 *If the control law and the update law are given as in eq(21) and in eq(24)-(25) then tracking errors will asymptotically converge to a neighborhood of zero.*

Proof:
Defining Z_s^* as supports of fuzzy set in fired rules, let

$$
{\Theta_s^\star}^T \xi_s(z_s) = (|\Delta Cs| + U(z_h))sgn(s) \quad as \quad z_s \in Z_s^*
$$

As $s_\Delta \neq 0$:

$$
\begin{aligned}
{\Theta_s^\star}^T \xi_s(z_s) &= |\Theta_s^\star|^T \xi_s(z_s) sgn(s) \\
{\Theta_s}^{*T} \xi_s(z_s) &= |\Theta_s^*|^T \xi_s(z_s) sgn(s)
\end{aligned}
$$

Referring to Proposition 1, let c_s is denoted as an $n \times 3n$ matrix, and its element is defined as $c_{s_{ik}} = sup_{z_s \in Z_s^C}[\left|\frac{\partial g_i(z_s)}{\partial z_{s_k}}\right|]$, where $g = \Delta Cs + \epsilon_h$, and

$$
\Delta_s(z_s) = [Z_s^{c_1+1} - Z_s^{c_1}, \cdots, Z_{h_{3n}}^{c_{3n}+1} - Z_{s_{3n}}^{c_{3n}}]^T,
$$

then

$$|\Theta_s^*| \leq |\Theta_s^*| + c_s \Delta_s. \tag{26}$$

Define

$$\begin{aligned}\Phi_h &= \Theta_h^* - \Theta_h \\ \Phi_s &= \Theta_s^* - \Theta_s\end{aligned}$$

dynamical equation with sliding mode form :

$$\begin{aligned}M\dot{s} &= -Ks - Cs + d(t)(\epsilon_h + \Delta C s {\Phi_h}^T \xi_h(z_h) \\ &\quad + {\Phi_s}^T \xi_s(z_s) - \Theta_s^*)\xi_s(z_s) + (1 - d(t)) \\ &\quad (h + \Delta Cs - U_s(z_h) sgn(s))\end{aligned}$$

Define a Lyapunov function

$$V = \frac{1}{2} s_\Delta^T M s_\Delta + \frac{1}{2r}(Tr({\Phi_h}^T \Phi_h) + Tr({\Phi_s}^T \Phi_s)) \tag{27}$$

$$\begin{aligned}\dot{V} &= \frac{1}{2} s_\Delta^T (\dot{M} - 2C) s_\Delta + s_\Delta^T M \dot{s}_\Delta \\ &\quad + \frac{1}{r}(Tr(\dot{\Phi}_h^T \Phi_h) + Tr(\dot{\Phi}_s^T \Phi_s)) \\ &\leq -s_\Delta^T K s_\Delta + d(t) s_\Delta^T (\epsilon_h + \Delta Cs - {\Theta_s^*}^T \xi_s(z_s)) \\ &\quad + (1 - d(t)) s_\Delta^T (-U_s(z_s) sgn(s) + h + \Delta Cs) \\ &\leq -{s_\Delta}^T K s_\Delta \end{aligned} \tag{28}$$

Using Barbala eq Lemma, it is straight forward to show that $s_\Delta \rightarrow 0$ as $t \rightarrow \infty$. This result implies that tracking errors are bounded [5].

3.3 Two-Stage Design

In this subsection, considerations of hardware limitations are combined with previous algorithm in subsection 3.2. From the analysis of subsection 3.2 , it is very clear that the last tracking errors depend on ranges of the deadzone, and our objective is either to raise higher precision or to reduce the total number of rules. Hence, in addition to the optimal parameters matrices, optimal regressive vectors also need to be considered. Therefore, an efficient arrangement for elements of supports of fuzzy set is main work in this subsection.

In general, a point support set of the relevant fuzzy sets is formed through two ways: way(1) is a data table; way(2) is a function form. For the former, the data set has to be memorized. This leads to extra memory-space needed. It even costs more computation time to find irregular address of rules. Oppositely, the latter did not face the previous difficulties, but it seems to be very hard for finding a suitable function form. An alternative is to combine way(1) with way(2). Hence, we add a fuzzy rule-base before previous control algorithm. This architecture is shown in Figure 2, where a ramp function is used to determine the original supports of fuzzy sets. Apparently, it is very intuitive that s is large, then Δ_s is large ; s is small, then Δ_s is small. This concept lays down rule space in efficiency and raises the higher precision because of the smaller deadzone range. Hence, the fuzzy rule-base is designed as follows:

$$\begin{array}{llllllll} R[i]: & & \text{If} & s_1 & \text{is} & {S_1}^{(i)} & \text{then} & y_{s_1} = Y_{s_1}^{(i)} \\ & or & \text{If} & s_2 & \text{is} & {S_2}^{(i)} & \text{then} & y_{s_2} = Y_{s_2}^{(i)} \\ & \vdots & & & & & & \\ & or & \text{If} & s_n & \text{is} & {S_n}^{(i)} & \text{then} & y_{s_n} = Y_{s_n}^{(i)} \end{array} \tag{29}$$

where $y_s = [y_{s_1}, \cdots, y_{s_n}]^T$ is denoted as a fuzzy output vector and $Y_{s_j} = \{Y_{s_j}^1, Y_{s_j}^2, \cdots, Y_{s_j}^\beta, \cdots, Y_{s_j}^{l_j}\}$ is denoted as a support of the fuzzy set, where $Y_{s_j}^\beta$ is an element of the support set Y_{s_j}. When fuzzy rule base is designed completely, furthermore, we let y_s be a fuzzy input vector, another support of fuzzy set with ramp function form is given as follows:

$$Y_{s_j}^{(new)} = \{1, 2, \cdots, R_{s_j}\}. \tag{30}$$

Therefore, memory-space of the data set may be condensed as $l_1 + l_2 \cdots + l_n \ll R_{s_1} \times R_{s_2} \cdots \times R_{s_n}$.

4 Simulation 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 Figure 3. A simulation is run in assumption that gravity vector of the arm is unknown. To show the effectiveness of adaptive robust law, τ_s, we will neglect the compensation function τ_h. The same desired trajectory $45\sin(\pi t)$ 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 is $5 \times r_q \times r_s = 5 \times 41 \times 41$. Simulation is run by 4-order Runge-Kutta method and results are listed in Figure 4. At the beginning, we only use *PD* controller to compensate for the uncertainties in the first two periods of sine wave, then incorporated the ARFC, right after the application of ARFC , the tracking errors are quickly driven toward zero. This shows that ARFC has a fast converging feature.

5 Conclusions

In this paper, though the efforts had been made in reducing memory-space, the number of rules is still very large, especially, as n is large. Hence, robust controller design plays an important role in the case in which fuzzy rule-base is not sufficient to approximate the uncertainty of the controlled system. Further investigation on how to reduce memory-space usage, such as decentralized design or other intelligent methods will be considered in the future work.

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 Direcet 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.*

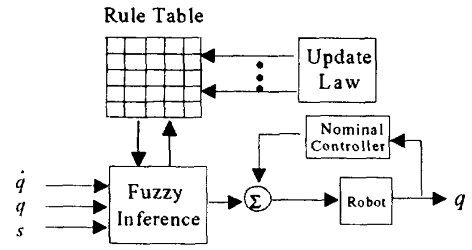

Fig 1. The architecture of ARFC

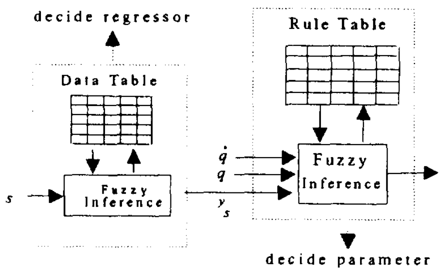

Fig. 2. The architecture of two-stage design

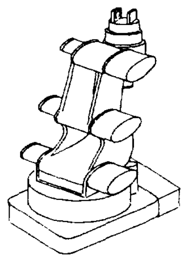

Fig. 3. A type robot

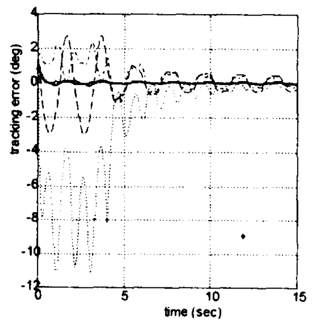

Fig. 4. The simulation result