Proceedings of the 1996 IEEE
International Conference on Robotics and Automation
Minneapolis, Minnesota - April 1996

A Neuro-Fuzzy Logic Controller for Trajectory Tracking of Uncertain Robots

Chih-Hsin Tsai [†] Jing-Sin Liu [*] and Wei-Song Lin [†]
[*] Institute of Information Science, Academia Sinica
Nankang, Taipei, Taiwan 115, R.O.C.
[†] Department of Electrical Engineering, National Taiwan
University, Taipei, Taiwan 106, R.O.C.

Abstract This paper presents an adaptive fuzzy computed-troque controller, that enhances fuzzy controllers with an embedded adjustable two-stages credit assignment and self-learning capability, for uncertain robots, to on-line track a prescribed trajectory. An adaptation law for the parameters of controller is combined with the dead-zone technique to guarantee a given attenuation region of tracking error in the presence of torque disturbance. Simulations of a two-link robot carrying a heavy load illustrate the effectiveness and attenuation capability of the controller for on-line trajectory tracking in the presence of inertial parameters uncertainties and torque disturbances.

1. Introduction

Motion control of uncertain robots has attracted a lot of researches (see e.g. [1]) and a lot of tracking controller design methods, ranging from traditional computed-torque controller, independent joint PID controller, robust or variable structures controllers [7], [8], model-based or parameter adaptive controllers to more recently intelligent controllers (such as iterative learning controllers [11], radial basis, multilayered, feedback-error-learning neural networks [4], [9], [16], [17] or fuzzy logic [4] based controllers) have been proposed to treat the robot tracking problems of varying degree requirements. Advances in the robot controllers design are mainly for developing methods that could adapt to variations of robot dynamics and payloads to improve performance for tracking the desired trajectory as closely as possible over a wide range of configurations. Previous work has shown that certain physical properties, e.g. passivity property, linear-in-parameter property, skew-symmetric property of robot dynamics are of use in designing robust and adaptive robot controllers and in proving the stability. However, in the presence of unstructured uncertainties or torque disturbance, the adaptive approaches can't guarantee the tracking performance or even the stability [6] due to their sensitivity of robot dynamics properties. In addition, to exploit the nonlinear mapping capability of neural networks to learn the inverse dynamics of uncertain robots, neural network based robot controllers require a tedious training phase for a given trajectory. Moreover, their generalization capability for different trajectories is not fully investigated. Also there are difficulties in choosing appropriate testing trajectories which contain sufficient information for learning the inverse dynamics. Only local stability around the operating points is ensured for tracking error dynamics.

On the other hand, fuzzy mehtods provide an efficient way to cope with uncertainties. The goal of this paper is thus to construct for uncertain robots a learnable controller–an adaptive fuzzy control system–to guarantee the tracking performance: achieve satisfactory on-line trajectory tracking in the presence of imperfect knowledge by meeting the requirement that the tracking error is attenuated to a prescribed region. Our approach is to employ fuzzy technique which could implement on-line computed-torque control that computes the necessary torque to move along a given desired trajectory, without detailed information of robot dynamics.

The organization of the paper is as follows. In Section 2, a fuzzy logic implementation of computed-torque control for robot motion tracking is presented. To train the controller network, its parameters are tuned. A modified update law for parameters using dead-zone technique is given in Section 3. The tracking performance and stability of the closed-loop robot system are investigated in Section 4. Simulations of a two-link robot carrying a heavy payload are given to illustrate the controller performance in Section 5. Section 6 is the conclusion.

2. Uncertain Robot Dynamics

The dynamics of n-link robots is expressed as [1]

$$M(q)\ddot{q} + c(q,\dot{q}) + g(q) = u + d(q,\dot{q},t) \tag{1}$$

with $q \in R^n$ joint variable, $M(q)$ the symmetric positive- definite inertia matrix, $c(q,\dot{q})$ the Coriolis and centrifugal forces, $g(q)$ the gravitational force, and u is the input torque. The combined effect of friction and external torque disturbance is represented by $d(q,\dot{q},t)$.

(1) can be put in the form

$$\ddot{q} = f(x) + G(x)u + v(x,t) \tag{2}$$

with $x^T = (q^T, \dot{q}^T), f = M^{-1}(-c-g), G = M^{-1}, v = M^{-1}d$.

Given a desired trajectory $q_d(t) \in R^n$, denote the ith joint tracking error state as $e_i = [q_i - q_{id}, \dot{q}_i - \dot{q}_{id}]^T$ and let $e = [e_1^T, \cdots, e_n^T]^T$. The robot motion tracking problem treated here is to design an adaptive fuzzy controller for uncertain robots such that e is attenuated to a given region in the presence of lumped uncertainties and bounded torque disturbances. For this purpose, this paper presents an adaptive fuzzy system for on-line learning the computed-torque control to guarantee the tracking performance. This is presented in Fig.1. Details about the design are described in the following section.

0-7803-2988-4/96 $4.00 © 1996 IEEE

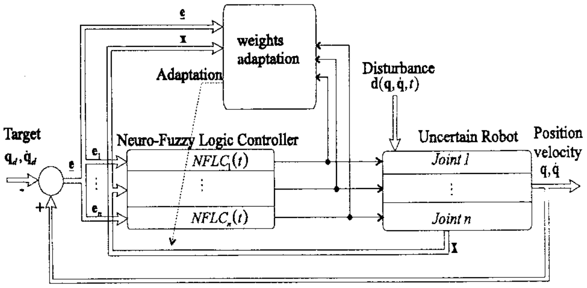

Figure 1

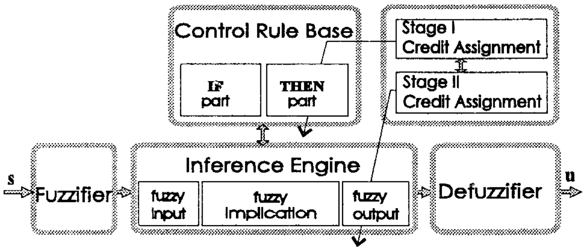

Figure 2

3. A Fuzzy Controller for Trajectory Tracking

As a preliminary, we introduce basic fuzzy concepts [10], [12]. Let $s = (s_1, \cdots s_{2n})$ represents the input (vector e or x) of the fuzzy system. Denote $R_{s,i}$ the fuzzy if-then rules for the ith joint of the robots. In general, the jth rule in the ith knowledge rule base, either $R_{x,i}$ for robot position and velocity or $R_{e,i}$ for robot tracking error, of the fuzzy system is defined by a set of linguistic rules of the following form:

$$R^j_{s,i} : IF \;\; s_1 \;\; is \;\; A^j_{i,1} \;\; AND \cdots AND \;\; s_{2n} \;\; is \;\; A^j_{i,2n} \\ THEN \;\; u_i \;\; is \;\; B^j_i. \tag{3}$$

where $A^j_{i,k}$ reference antecedent fuzzy set of s_k, and B^j_i reference consequent fuzzy set of the outputs of the fuzzy system. This set of fuzzy if-then rules forms a control rule base whose antecedent parts are related to the measurement and whose consequent parts determine the control action. The quality of the control action is inferred by a fuzzy inference engine and is evaluated by the credit assignments mechanism. Fig. 2 shows the concepts of the novel approximate reasoning fuzzy system embedded with adjustable two-stages credit assignment for fuzzy if-then rules (3).

3.1 Two-Stages Credit Assignment

The basic idea of rule credit assignment is to reward good rules by increasing their certainty of the consequent fuzzy set while punish bad ones by decreasing their certainty. There are two rule credit assignment stages presented in the fuzzy system of Fig. 2. First, at stage I, we reshape the consequent fuzzy set B^j_i of the original fuzzy rule base. This paper uses LR parametrization [12] as the consequent membership functions. Thus after stage I rule credit assignment, those membership functions become $\tilde{B}^j_i$.

By fuzzy implication inference, the corresponding output action (recommendation) of each rule is defined as [9], [13]

$$A'_i \circ R^j_{u,i}(s, u_i) := Sup_{u \in U}[A'_i(s) * I(A^j_i(s), \tilde{B}^j_i(u_i))] \tag{4}$$

where $A'_i(s)$ is an arbitrary fuzzy set input to the fuzzy system, $A^j_i(s) := A^j_{i,1}(s_1) * \cdots * A^j_{i,2n}(s_{2n})$ denotes the matching degree, $*$ is the T-norm [12], I the implication function and $\tilde{B}^j_i(u_i)$ is a reshaped $B^j_i(u_i)$ in original rule base (3) as a consequence of stage I credit assignment. On the other hand, the stage II credit assignment is imposed on the fuzzy output where we have determined the corresponding output action of each rule. Here, we refine them by giving a credit assignment, ω^j_{ii}, to the jth rule. Then the output fuzzy set becomes:

$$\omega^j_{ii} \cdot A'_i \circ R^j_i(s, u) \tag{5}$$

where "." is the multiplication operation.

3.2 A fuzzy computed-torque controller

Similar to the center-average defuzzification method [9], the defuzzification of a multi-input multi-output fuzzy system with credit assignment is defined as

$$u(t) = \begin{bmatrix} \Sigma_j \omega^j_{11} A^j_1 & \cdots & \Sigma_j \omega^j_{1n} A^j_1 \\ \vdots & \ddots & \vdots \\ \Sigma_j \omega^j_{n1} A^j_n & \cdots & \Sigma_j \omega^j_{nn} A^n_n \end{bmatrix}^{-1} \begin{bmatrix} \Sigma_j \omega^j_{11} A^j_1 \cdot \tilde{c}^j_{u,1} \\ \vdots \\ \Sigma_j \omega^j_{nn} A^j_n \cdot \tilde{c}^j_{u,n} \end{bmatrix} \tag{6}$$

where $\tilde{c}^j_{u,i}$ denotes

$$\tilde{c}^j_{u,i} \equiv the \;\; centroid \;\; of \;\; the \;\; set \{u : B^j_i(u) \geq A^j_i(s)\} \tag{7}$$

(using the LMOM method [5]), ω^j_{ii} represents the credit assigned to $R^j_{s,i}$ for joint i, while ω^j_{ik}, $i \neq k$, is used to counteract the dynamic interactions between the robot joints i and k.

Fig. 3 shows four different types of processing nodes in the network. Each corresponds, respectively, to a substage shown in Fig. 2:

- *Input Layer*: The two inputs to the fuzzy system are e_i, the tracking error of ith joint, and x, the position and velocity of the robot.
- *Rule Matching Layer*: Each node here obtains the rule matching degree $\hat{g}^j_i(s) = A^j_i(s)$. To make (6) a form of computed-torque control, two types of antecedent membership functions are used. For $s = x$, $A^j_{i,k}$ is the Gaussian membership function defined by

  $$A^j_{i,k}(x_k) = exp\left[-(x_k - c^j_{s,k})^2\right] \tag{8}$$

  For $s = e$, we choose a special class of bell-shape membership functions that satisfy

  $$\Sigma_j \hat{g}^j_i(e_i) = 1 \tag{9}$$

  where the summation is over the number of $R_{e,i}$ rules for ith joint.
- *Fuzzy Implication Layer*: Each node in this layer obtains a singleton implication fuzzy set and compute its location $\tilde{c}^j_{u,i}$ by

  $$\tilde{c}^j_{u,i} = c^j_{u,i} - \frac{1}{2}\beta^j_{ii}\alpha^j_i \left((\hat{g}^j_i)^{-1} - 1\right)^{1/2} \tag{10}$$

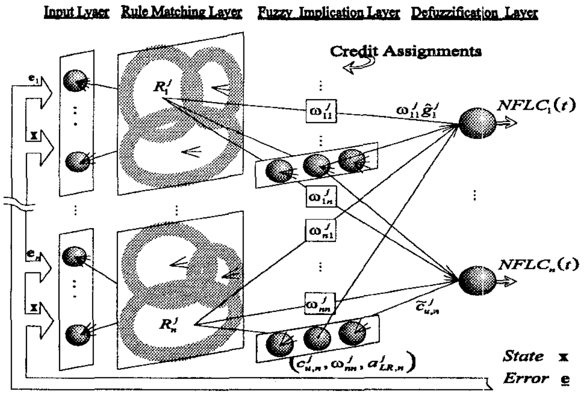

Figure 3

where $c_{u,i}^j$ is the center of A_i^j, $\beta_{i,i}^j$ the stage I credit of B_i^j $\alpha_i^j = \alpha_{L,i}^j - \alpha_{R,i}^j$, the difference of left and right spread of fuzzy membership function A_i^j. On the other hand, the R_e^i rules are chosen to be of Takagi-Sugeno type. Its consequent membership function B_i^j is a singleton with support represented as the form of synthesis input

$$c_{u,i}^j = \ddot{q}_{i,d} - a_i^{j^T} e_i, \tag{11}$$

where $a_i^j = (k_{pi}^j, k_{vi}^j)^T$. Suppose the credits of all these rules are assigned to be 1, then we have $\tilde{c}_{u,i}^j = c_{u,i}^j$.

- *Defuzzification Layer*: This layer contains n nodes which compute the on-line computed-torque control according to (6).

Let $a_i^j = a_i$ for all j in (11) and for convenience, we choose $\beta_{ii}^j = 1/\omega_{ii}^j$ such that the credits of all rules at joint i are assigned simultaneously in stages I and II and the number of adjustable parameters could be reduced. Accordingly, using (10), (11) and (12), the equation (6) resolves into the computed-torgue form

$$u(t) = D^{-1}\left(x, \theta^{(\omega)}\right)\left(F(x, \theta^{(ca)}) + h(e,t)\right) \tag{12}$$

where

$$D\left(x, \theta^{(\omega)}\right) = \begin{bmatrix} \omega_{11}\hat{g}_1(x) + c_{11} & \cdots & \omega_{1n}\hat{g}_1(x) + c_{1n} \\ \vdots & \ddots & \vdots \\ \omega_{n1}\hat{g}_n(x) + c_{n1} & \cdots & \omega_{nn}\hat{g}_1(x) + c_{nn} \end{bmatrix}$$

$$F(x, \theta^{(ca)}) = \begin{bmatrix} -\theta_1^{(ca)}\hat{f}_1 \\ \vdots \\ -\theta_n^{(ca)}\hat{f}_n \end{bmatrix}, h(e,t) = \begin{bmatrix} \ddot{q}_{1d} - a_1^T e_1 \\ \vdots \\ \ddot{q}_{nd} - a_n^T e_n \end{bmatrix}$$

with $\theta^{(\omega)} = (\theta_1^{(\omega)}, \cdots, \theta_n^{(\omega)})^T$, $\theta_1^{(\omega)} = (\omega_{i1}, \cdots, \omega_{in})^T$, $\omega_{ij} = (\omega_{ij}^1, \cdots, \omega_{ij}^m)^T$, $\hat{g}_i = (\hat{g}_i^1, \cdots, \hat{g}_i^m)^T$, $\theta^{(ca)} = (\theta_1^{(ca)}, \cdots, \theta_n^{(ca)})^T$, $\theta_i^{(ca)} = (\omega_{ii}^1 c_u^1, \cdots, \omega_{ii}^m c_u^m, \alpha_i^1, \cdots, \alpha_i^m)^T$, $\hat{f}_i^j = (\hat{f}_i^1, \cdots, \hat{f}_i^m, \tilde{f}_i^1, \cdots, \tilde{f}_i^m)^T$, with $\hat{f}_i^j = -\hat{g}_i^j$, and $\tilde{f}_i^j = \hat{g}_i^j((\hat{g}_i^j)^{-1} - 1)^{1/2}$ where m is the number of $R_{x,i}$ rules. In the above, $C := (c_{ij})$ is a bias matrix.

Remark: As a function approximator, the fuzzy system uses the term $\omega_{ij}\hat{g}_i + c_{ij}$ to learn the (i,j) element of matrix G, and the term $\theta_i^{(ca)^T}\hat{f}_i$ to learn f_i in vector f. It can therefore be easily seen that the fuzzy system tries to on-line learn a computed-torque control in the presence of uncertainties in robot dynamics.

4. Learning algorithm

Referring to the controller network in Fig.3, the connections linking from the second layer to the output layer and the parameters in the nodes of the third layer (i.e., $\omega_{ir}^j, c_{u,i}^j$ and α_i^j) are adjustable. Define $\theta_i = (\theta_i^{(ca)}, \theta_i^{(\omega)})$. Let the parameter θ_i be updated according to

$$\theta_i(t) = \theta_{0_i} + \Delta\theta_i(t) \tag{13}$$

where θ_{o_i} is the initial parameter value.

4.1 Adaptation law

Plugging (13) into (2), subtracting $Du(t)$ and adding its identity $(F(x, \theta^{(ca)}) + h(e,t))$ to the right hand side of (2) we have

$$\ddot{q}_i = \ddot{q}_{id} - a_i^T e_i + \left(f_i - \theta_i^{(ca)^T}\hat{f}_i\right) + \Sigma_{j=1}^n\left(g_{ij} - \omega_{ij}^T\hat{g}_i - c_{ij}\right)u_j \tag{14}$$

Substituting (3) into (4) and after some manipulations, the tracking error equation for *ith* joint can be written as

$$\dot{e}_i = A_i e_i - b_i w_i^T \Delta\theta_i + b_i\xi_i \tag{15}$$

where

$$A_i = \begin{bmatrix} 0 & 1 \\ -k_{pi} & -k_{vi} \end{bmatrix}, b_i = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, w_i = \begin{bmatrix} \hat{f}_i \\ w_i^{(\omega)} \end{bmatrix},$$

$$\xi_i = (f_i - \theta_{0_i}^{(ca)^T}\hat{f}_i) + \Sigma_{j=1}^n(g_{ij} - \omega_{o_{ij}}^T\hat{g}_i - c_{ij})u_j + v_i \tag{16}$$

with $w_i^{(\omega)} = \hat{g}_i u$.

In this paper,an adaptation law for tuning the parameters of fuzzy controller for each joint is given by

$$\Delta\theta_i = R_i^{-1} w_i b_i^T P_i e_i \tag{17}$$

where $P_i = P_i^T > 0$ is the solution of the following Riccati-like equation

$$P_i A_i + A_i^T P_i + \frac{1}{\rho^2}P_i b_i b_i^T P_i + Q_i = 0 \tag{18}$$

with $Q_i > 0, \rho > 0$ and

$$R_i = Block\ diag\ (R_i^{(c)}, R_i^{(a)}, R_i^{(\omega)}) \tag{19}$$

$$R_i^{(\omega)} = Block\ diag\ (R_i^{(1)}, \cdots, R_i^{(n)}) \tag{20}$$

4.2 A dead-zone modification of adaptation law

A continuous version of (7) could be expressed as

$$\dot{\theta}_i = R_i^{-1} w_i b_i^T P_i e_i \tag{21}$$

In practice, bounds on the components of the desired parameters vector are generally known, so that unboundedness of vector $\theta_i(t)$ can be avoided by suitably modifying the adaptation policy. Furthermore, to counteract the modeling error and the parameter estimation errors, a deadzone of size d_0 is employed in the adaptation law. Let $\theta_{i\perp} = \theta_i/|\theta_i|$. Define $P = Block\ diag\ (P_1, \cdots, P_n)$, and $B = Block\ diag\ (b_1, \cdots, b_n)$. Suppose $\theta_i(t)$ is required to be inside a set M_{θ_i}, then the modified adaptation law [13] is : $\dot{\theta}_i = 0$, for $e^T P b b^T P e \le d_0^2$, otherwise we have

$$\dot{\theta}_i = (I - d_M(\theta_i)\theta_{i\perp}\theta_{i\perp}^T)R_i^{-1} w_i b_i^T P_i e_i \tag{22}$$

where the distance measure $d_M(\theta_i) = 0$ for $\theta_i^T(R_i^{-1} w_i b_i^T P_i e_i) \leq 0$,
otherwise

$$d_M(\theta_i) = min\,[1, dist(\theta_i, M_{\theta i})/\varepsilon^*], \varepsilon^* > 0. \tag{23}$$

4.3 Adaptation law for symmetric G

First, note that the lower component of $\dot{\theta}_i$ is $\dot{\theta}_i^{(\omega)}(t) = 0$ for $e^T Pbb^T Pe \leq d_0^2$, otherwise we have

$$\dot{\theta}_i = (I - d_M(\theta_i)\theta_{i\perp}\theta_{i\perp}^T)R_i^{(\omega)^{-1}} w_i^{(\omega)} b_i^T P_i e_i \tag{24}$$

Since G is symmetric, we let $\omega_{ij} = \omega_{ji}$ in the network. In this case, we modify the continuous adaptation law as follows: The adaptation algorithm for symmetric G replaces $\dot{\theta}_i^{(\omega)}$ of (27) by another $\dot{\phi}_i^{(\omega)}$, where

$$\dot{\phi}_i^{(\omega)} := ith\ row\ of\ \frac{1}{2}\left(\dot{\theta}^{(\omega)} + \dot{\theta}^{(\omega)^T}\right). \tag{25}$$

4.4 Selection of adaptation gains

It is interesting to observe that the matrices $(R_i^{(c)}, R_i^{(a)})$, $R_i^{(\omega)}$ represent the inverse of adaptation gains of $\theta_i^{(ca)}, \theta_i^{(\omega)}$, respectively. Since for robot dynamics the variation of the components of G is usually smaller than those of f, the adaptation gains of $\theta_i^{(\omega)}$ (and thus $\phi_i^{(\omega)}$) could be smaller than that of $\theta_i^{(ca)}$. This provides us a guideline to choose controller parameters.

5. Guaranteed tracking performance

Assume that there exist parameters $\theta_1^*, \cdots, \theta_n^*$ such that satisfactory approximation accuracy of on-line computed-torque input can be achieved, or

$$max_x \| f(x) - F(x, \theta^{(ca)*}) \|_2 \leq \varepsilon_f$$

$$max_x \| G(x) - D\left(x, \theta^{(\omega)^*}\right) \|_2 \leq \varepsilon_g.$$

Define the parameters estimation error as $\tilde{\theta}_i = \theta_i - \theta_i^*$. Then the error equation (5) can be rewritten as

$$\dot{e}_i = A_i e_i - b_i w_i^T \tilde{\theta}_i + b_i \xi_i \tag{26}$$

where the coupling effects between robot joints and the combined effect of approximation error and disturbance are

$$\xi_i = \left(f_i - \theta_i^{(ca)^T} \hat{f}_i\right) + \Sigma_{j=1}^n \left(g_{ij} - \omega_{ij}^{*^T} \hat{g}_i(x) - c_{ij}\right) u_j + v_i \tag{27}$$

5.1 Feasibility of controller

Denote $\tilde{\theta}^{(\omega)} = \theta^{(\omega)} - \theta^{(\omega)^*}$. In the case there exist ε and δ small enough such that $\varepsilon_g \leq \varepsilon$ and $\|\tilde{\theta}^{(\omega)}\|_2 \leq \delta$, we can show that $D\left(x, \theta^{(\omega)}\right)$ exists, which in turn, guarantees the feasibility of computed-torque controller (12). Let $\|G(x)\|_2 \geq g > 0$ in robot dynamics. Since

$$\|D\left(x, \theta^{(\omega)}\right) - G(x)\|_2 \leq \delta + \varepsilon$$

then we have

$$\begin{aligned}
&\|G^{-1}(x)\left(D(x, \theta^{(\omega)}) - G(x)\right)\|_2 \\
&\leq \|G^{-1}(x)\|_2 \|D(x, \theta^{(\omega)}) - G(x)\|_2 \\
&\leq (1/g)(\delta + \varepsilon) \\
&\leq 1
\end{aligned}$$

if δ and ε are small enough. Then, using matrix properties [2] we have $I_n + G^{-1}(x)(D(x, \theta^{(\omega)^*})G(x))$ is invertible which implies that $G(x)(I_n + G^{-1}(x)(D(x, \theta^{(\omega)^*}) - G(x))) = D(x, \theta^{(\omega)^*})$ is invertible.

5.2 Initial parameters for better tracking performance

In practice, nominal knowledge about the manipulator dynamics is available. Hence, an "approximation" to $G(x)$, denoted by $G_0(x|nominal\ robot\ parameters)$, is known a priori. This knowledge could be employed to satisfy the accuracy requirement of δ small enough, by selecting a suitable initial parameters $\theta_0^{(\omega)}$. This is done here by using the least square (LS) algorithm [14]. Our approach is based on the element-by-element minimization of the approximation error through a sufficient number of training data $\{x^{(k)}\}$:

$$\begin{aligned}
E(\omega_{0ij}) &= \Sigma_k \mid (G_0(x^{(k)}|nominal\ robot\ parameters) \\
&\quad - D(x^{(k)}, \theta^{(\omega)}))_{ij} \mid^2
\end{aligned} \tag{28}$$

where the set $\{x^{(k)}\}$ contains either the sampled feature points on or the points close to the desired trajectory. Furthermore, the same technique can also be used to satisfy the approximation accuracy requirement for f.

In addition, since the inertia matrix (thus G matrix) is lower and upper bounded [1], the bias matrix C can be selected with an aim to meet the bounds requirement of G in the process of LS minimization.

5.3 Tracking performance

The following shows the tracking performance of adaptive fuzzy controller for uncertain robot manipulators.

Theorem. Consider the robots (2) with unknown but bounded $f_i(x)$, $v_i(x,t)$ and $g_i(x) \neq 0, i = 1, \cdots, n$. Assume that the controller (12) is adopted with the adaptation policy (24). Then in the bounded state space $x \in \Omega = \{x : \|x\| \leq \gamma\}$, we have θ_i and the control input are bounded. Let $\bar{\xi} = (\xi_1, \cdots, \xi_{\bar{n}})^T$, $Q = Block\ diag\ (Q_1, \cdots, Q_n)$ and assume that there exists $\xi = \sup_{\theta_i, x, t} \Sigma_i \|\xi_i(\theta_i, x, t)\|^2$, then e converges to the residual set $\{e : e^T Qe \leq \rho^2 \xi$ or $e^T Pbb^T Pe \leq d_0^2\}$. Moreover, for ε_f and ε_g small enough such that $\xi \leq d_0/2\rho^2$, then e converges to the deadzone $\{e : e^T Pbb^T Pe \leq d_0^2\}$.

Proof: Let V_θ be a positive-definite function of the form

$$V_\theta = \frac{1}{2}\Sigma_{i=1}^n \theta_i^T \theta_i$$

$\dot{V}_\theta = \Sigma_{i=1}^n \theta_i^T \dot{\theta}_i \leq 0$. Therefore, we can guarantee the boundedness of θ_i. Thus the arguments θ_i, x in (12) are bounded, in view of the assumption of $x \in \Omega$. So we have u bounded.

Next, for the whole error system, define the Lyapunov function candidate as $V = V_1 + V_2 + \ldots + V_n$ where for each joint we define the Lyapunov function

$$\begin{aligned}
V_i &= \tfrac{1}{2}d_0^2 + \tfrac{1}{2}\tilde{\theta}_i^T R_i \tilde{\theta}_i, & if\ \ e^T Pbb^T Pe \leq d_0^2 \\
&= \tfrac{1}{2}e_i^T P_i e_i + \tfrac{1}{2}\tilde{\theta}_i^T R_i \tilde{\theta}_i, & otherwise
\end{aligned} \tag{29}$$

Taking the time derivative of V_i, we readily obtain $\dot{V}_i = 0$ for $e^T Pbb^T Pe \leq d_0^2$, otherwise we have

$$\begin{aligned}
\dot{V}_i &= e_i^T P_i \dot{e}_i + \tilde{\theta}_i^T \left(I - d_M(\theta_i)\theta_{i\perp}\theta_{i\perp}^T\right) w_i b_i^T P_i e_i \\
&= \frac{1}{2}e_i^T \left(A_i^T P_i + P_i A_i\right) e_i + \xi_i b_i^T P_i e_i \\
&\quad - d_M(\theta)\tilde{\theta}_i^T \theta_{i\perp}\theta_{i\perp}^T w_i b_i^T P_i e_i
\end{aligned}$$

In the case $d_M(\theta_i) \neq 0, \theta_i$ will be in the exterior of $M_{\theta i}$ and $\theta_i^T(R_i^{-1} w_i b_i^T P_i e_i) > 0$ which implies $\theta_{i\perp}^T(R_i^{-1} w_i b_i^T P_i e_i) > 0$. Suppose M_{θ_i} is appropriately selected such that θ_i^* is in the interior of M_{θ_i}, then we have

$$\begin{aligned} \tilde{\theta}_i^T \theta_{i\perp} &= \theta_i^T \theta_{i\perp} - \theta_i^{*T} \theta_{i\perp} \\ &= \tfrac{1}{2} \mid \theta_i \mid \left((\theta_i^* - \theta_i)^2 + \theta_i^2 - \theta_i^{*2} \right) > 0 \end{aligned} \tag{30}$$

Thus,

$$\begin{aligned} \dot{V}_i &\leq \tfrac{1}{2} e_i^T (A_i^T P_i + P_i A_i) e_i + \xi_i b_i^T P_i e_i \\ &\leq -\tfrac{1}{2} e_i^T Q_i e_i + \tfrac{1}{2} \rho^2 \mid \xi_i \mid^2 \end{aligned} \tag{31}$$

so we have

$$\dot{V} \leq -\frac{1}{2} e^T Q e + \frac{1}{2} \rho^2 \xi \tag{32}$$

Thus e converges to the residual set $\{e : e^T Q e \leq \rho^2 \xi \ or \ e^T P b b^T P e \leq d_0^2\}$.

Finally, consider the case that ε_f and ε_g are small enough such that $\xi \leq d_0/2\rho^2$ for $e^T P b b^T P e > d_0^2$. Thus

$$\begin{aligned} \dot{V} &= -\frac{1}{2} e^T Q e - \frac{1}{2} \rho^2 e^T P b b^T P e + \xi b^T P e \\ &\leq -\frac{1}{2} e^T Q e - \left(\frac{1}{2} \rho^2 \| e^T P b \| - \xi \right) \| b^T P e \| \\ &\leq -\frac{1}{2} e^T Q e \end{aligned} \tag{33}$$

Then, following the arguments of [16] we can conclude that e converges to the deadzone $\{e : e^T P b b^T P e \leq d_0^2\}$.

Q.E.D.

Corollary. Consider the robot (2) with a symmetric G matrix. Then all properties described in Theorem also hold for the robot controller (12)-(25).

Proof: Note that the derivative of Lyapunov function (29) can be decomposed as

$$\dot{V}_i = e_i^T P_i \dot{e}_i + \tilde{c}_{u_i}^T R_i^{(c)} \dot{c}_{u_i} + \tilde{a}_i^T R_i^{(a)} \dot{a}_i + \tilde{\theta}_i^{(\omega)^T} R_i^{(\omega)} \dot{\theta}_i^{(\omega)} \tag{34}$$

Substituting (25) into (4), we have

$$\dot{V}_i = e_i^T P_i \dot{e}_i + \tilde{c}_{u_i}^T R_i^{(c)} \dot{c}_{u_i} + \tilde{a}_i^T R_i^{(a)} \dot{a}_i + \tilde{\theta}_i^{(\omega)^T} R_i^{(\omega)} \dot{\phi}_i^{(\omega)} \tag{35}$$

In view of the properties: (i)$\tilde{\omega}_{ij} = \tilde{\omega}_{ji}$, due to $\omega_{ij} = \omega_{ji}$, (ii)$R_i^{(j)} = R_j^{(i)}$, one can verify that

$$\Sigma_{i=1}^n \tilde{\theta}_i^{(\omega)^T} R_i^{(\omega)} \dot{\theta}_i^{(\omega)} = \Sigma_{i=1}^n \tilde{\theta}_i^{(\omega)^T} R_i^{(\omega)} \dot{\phi}_i^{(\omega)} \tag{36}$$

Thus,

$$\Sigma_{i=1}^n \dot{V}_i \mid_{along(25)} = \Sigma_{i=1}^n \dot{V}_i \mid_{along(24)} \tag{37}$$

so that (32) and (33) holds.

Q.E.D.

6. Simulation

Simulation is done for a two-link robot manipulator with dynamics [9]:

$$\begin{aligned} &((m_1 + m_2) r_1^2 + m_2 r_2^2 + 2 m_2 r_1 r - 2cos(q_2) + J_1) \ddot{q}_1 \\ &+ (m_2 r_2^2 + m_2 r_1 r_2 cos(q_2)) \ddot{q}_2 - m_2 r_1 r_2 sin(q_2)(\dot{q}_1 + \dot{q}_2) \\ &+ (m_1 + m_2) l_1 cos(q_2) + m_2 l_2 cos(q_1 + q_2) = u_1 + d_1 \end{aligned}$$

$$\begin{aligned} &(m_2 r_2^2 + m_2 r_1 r_2 cos(q_2)) \ddot{q}_1 + (m_2 r_2^2 + J_2) \ddot{q}_2 \\ &+ m_2 r_1 r_2 sin(q_2) \dot{q}_2 + m_2 l_2 cos(q_1 + q_2) = u_2 + d_2 \end{aligned}$$

where the combined effects of friction and the external torque disturbance are

$$d_1 = 2.0 sin(\dot{q}_1) + 2.5 sin(\dot{q}_2) + 0.5 sin(t)$$

$$d_2 = 5.0 sin(\dot{q}_1) + 4.0 sin(\dot{q}_2) + 0.4 sin(t)$$

and $l_1 = 2$, $l_2 = 1.6$, $r_1 = 0.5 l_1$, $r_2 = 0.5 l_2$, $J_1 = J_2 = 5$, $m_1 = 0.5$, $m_2 = 6.35$. The excessive ratio between m_1 and m_2 is to emphasize the payload effect.

For each joint position or velocity, only four linguistic labels $\{NB,\ NS,\ PS,\ PB\}$ are used with membership functions

$NB : f(r) = 1/(1+exp(5(r+1)))$, $NS : f(r) = exp(-(r+0.6)^2)$,

$PS : f(r) = exp(-(r-0.6)^2)$, $PB : f(r) = 1/(1+exp(-5(r-1)))$.

where r is the joint position or velocity. Moreover, we let $a_1^j = a_2^j = 0$ for all j, i.e., the left and right spreads of the consequent membership function are equal in the adaptation period. We consider there are no fuzzy control rules. We set bias $c_{ij} = 1$ for all i, j. The initial parameters $c_{u_1}(0)$ and $c_{u_2}(0)$ are chosen randomly in the interval (-50, 50). To obtain a set of appropriate initial parameters $\theta_0^{(\omega)}$, in (28) we use 32 testing points on the desired trajectories over one period [0,8]. The following nominal inertia parameters are used in LS minimization: $J_1^0 = 4.8$, $J_2^0 = 5.1$, $m_1^0 = 0.48$, $m_2^0 = 6.30$.

The other controller parameters are chosen as $Q_1 = Q_2 = 10 I_2$, $\varepsilon^* = 0.05$, $\alpha_{1,1} = \alpha_{2,1} = 100$, $\alpha_{1,2} = \alpha_{2,2} = 65$ and

$$R_1 = Block\ diag\,(0.005 I_{256}, 3000 I_{256}, 2000 I_{256}),$$

$$R_2 = Block\ diag\,(0.0025 I_{256}, 2000 I_{256}, 3000 I_{256}).$$

Two cases with different ρ values are simulated for

$$q_{1d}(t) = \pi/12 sin(0.5\pi t)$$

$$q_{2d}(t) = 2.5\pi/12 \cos(0.5\pi t) + 2.5\pi/24 \cos(0.25\pi t).$$

with the initial state $q_1(0) = -1.5$, $q_2(0) = -1.2$, $\dot{q}_1(0) = \dot{q}_2(0) = 0$. They are compared, with the responses shown in Fig. 4((a) position error (b) velocity error (c) control input) where solid line and dotted line denote $\rho_1 = \rho_2 = 0.01$ and $\rho_1 = \rho_2 = 0.5$, respectively. The tracking errors for the last joint are relatively small, whereas larger tracking error are observed in the first joint. In both simulation cases, the tracking errors are gradually within a tolerable accuracy region as the learning progresses.

7. Conclusion

For trajectory tracking of uncertain robotic manipulators, we have proposed a learnable controller for on-line learning of the computed-torque control to adapt to changing environment characterized by imperfect knowledge. The controller consists of a fuzzy system embedded with adjustable credit assignments and supervised by the desired trajectory. A combination of controller parameter adaptation law with a dead-zone technique is used to guarantee the stability and the attenuation of tracking error to a prescribed region. Simulations of a two-link robot manipulator have demonstrated that the controller is efficient in achieving satisfactory tracking accuracy in the presence of significant uncertainties. Simulations also show how a priori knowledge about the robot dynamics can be applied in an auxiliary manner to speed up the learning process and to improve the tracking accuracy.

References

[1] F.L. Lewis, C.T. Abdallah, and D.M. Dawson, *Control of Robot Manipulators.* New York: Macmillan, 1993.

[2] R.A. Horn, and C.R. Johnson, *Matrix Analysis.* Cambridge, U.K.: Cambridge University Press, 1985.

[3] C.M. Lim and T. Hiyama, "Application of fuzzy logic control to a manipulator," *IEEE Trans. Robotics and Automation*, vol.7, no.5, pp.688-691, 1991.

[4] D.T. Pham and S.J. Oh, "Adaptive control of a robot using neural networks," *Robotica*, vol.12, pp.553-561, 1994.

[5] H.R. Berenji and P.S. Khedkar, "Learning and tuning fuzzy logic controllers through reinforcements," *IEEE Trans. Neural Networks*, vol.3, no.5, pp.724-740, 1992.

[6] J.S. Reed and P.A. Ioannou, "Instability analysis and robust adaptive control of robotic manipulators," *IEEE Trans. Robotics and Automation*, vol.5, no.3, pp.381-386, 1989.

[7] J. Wang, W.N. Bailey, and S.J. Dodds, "A new sliding mode approach to the robust control of robotic manipulators with dynamic uncertainties," in *Proc. IEEE Conf. Robotics and Automation*, San Diego, California, pp.2100-2105, 1994.

[8] K.D. Young, "Controller design for a manipulator using theory of variable structure systems," *IEEE Trans. Systems, Man, and Cybernetics*, vol.8, pp.101-109, 1978.

[9] M. Kemal Cliz, "Stability and convergence of neurologic model based robotic controllers," in *Proc. IEEE Conf. Robotics and Automation*, Nice, France, pp.2051-2056, 1992.

[10] L.X. Wang, *Adaptive Fuzzy Systems and Control.* NJ: Prentice-Hall, 1994.

[11] S. Arimoto, S. Kawamura and F. Miyazaki, "Bettering operation of robots by learning," *J. Robotic Syst*, vol.1, pp.123-140, 1984.

[12] G.J. Klir and B. Yuan, *Fuzzy Sets and Fuzzy Logic.* NJ: Prentice-Hall, 1995.

[13] G. Kreisselmeier, and B.D.O. Anderson, "Robust model reference adaptive control," *IEEE Trans. Automatic Control*, vol.31, pp.127-134, 1986.

[14] D.G. Luenberger, *Linear and Nonlinear Programming.* Reading, MA: Addison-Wesley, 1984.

[15] F.C. Chen, and C.C. Liu, "Adaptively controlling nonlinear continuous-time systems using multilayer neural networks," *IEEE Trans. Automatic Control*, vol.39, no.6, pp.1306-1310. Jun. 1994.

[16] T. Ozaki, T. Suzuki, T. Furuhashi, S. Okuma and Y. Uchikawa, "Trajectory control of robotic manipulators using neural networks," *IEEE Trans. Industrial Electronics*, vol.38, no.3, pp.195-202, 1991.

[17] H. Miyamoto, M. Kawato, T. Setoyama and R. Suzuki, "Feedback-error-learning neural networks for trajectory control of a robotic manipulator," *Neural Networks*, vol.1, pp.251-265, 1988.

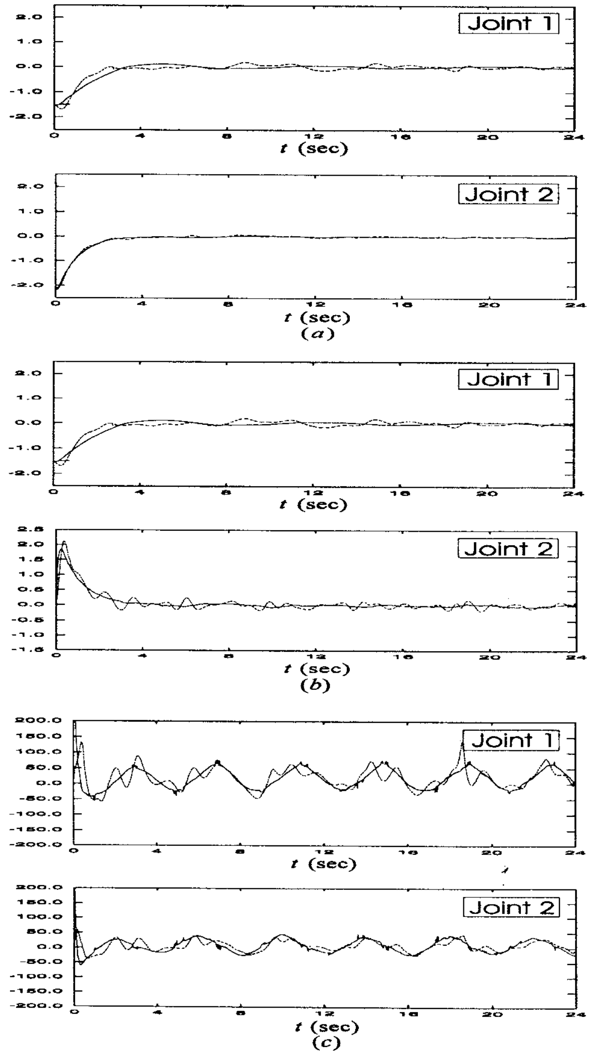

Figure 4