Proc. Instn Mech. Engrs Vol. 217 Part I: J. Systems and Control Engineering

Fuzzy and neuro-fuzzy approaches to control a flexible single-link manipulator

B Subudhi and **A S Morris***
Department of Automatic Control and Systems Engineering, University of Sheffield, UK

Abstract: In this paper, new fuzzy and neuro-fuzzy approaches to tip position regulation of a flexible-link manipulator are presented. Firstly, a non-collocated, proportional-derivative (PD) type, fuzzy logic controller (FLC) is developed. This is shown to perform better than typical model-based controllers (LQR and PD). Following this, an adaptive neuro-fuzzy controller (NFC) is described that has been developed for situations where there is payload variability. The proposed NFC tunes the input and output scale parameters of the fuzzy controller on-line. The efficacy of the NFC has been evaluated by comparing it with a fuzzy model reference adaptive controller (FMRC).

Keywords: fuzzy logic, neural network, flexible link, manipulator, tip position

NOTATION

$\mathbf{A}, \mathbf{B}, \mathbf{C}$	system matrices
A_i	normalization constant for the ith mode shape
b_j, b_k	threshold values for neurons in the output and hidden layers
$c_i(k)$	centre of the ith output membership function at iteration k
$\mathbf{D}$	damping matrix
$e(t), \dot{e}(t)$	tip position and velocity errors
E_D	dissipative energy
E_h, E_l, E_m	kinetic energy of the hub, link and payload
E_N	error function
E_P	total potential energy
EI	flexural rigidity
f_j, f_k	sigmoidal activation functions for hidden and output neurons
F	generalized forces
$G(s)$	transfer function of the reference model
h	step size
$\mathbf{H}$	positive definite (or semidefinite) constant matrix
$\mathbf{I}$	identity matrix
I_h, I_b	hub of inertia and mass moment of inertia of the link
J	performance index of the LQR
$\boldsymbol{K}$	LQR gain vector
K_E, K_{CE}, K_U	scale factors
K_{EP}, K_{CEP}, K_{UP}	scale factors for the fuzzy inverse model
K_P, K_D	PD controller gain parameters
$\mathbf{K}_w$	stiffness matrix
l	length of link
$\mathscr{L}$	Lagrangian
m, m_p	mass of link and payload mass
$\mathbf{M}$	mass matrix
n_i, n_h, n_o	number of nodes in the input, hidden and output layers
n_m	number of finite modes
n_r	number of rules that have contributions
O_k	kth node in the output layer
$\mathbf{P}$	solution of the Riccati equation
$q_i(t)$	time-varying modal function
Q	generalized coordinates
$\mathbf{Q}_{LQ}$	positive definite (or semidefinite) constant matrix
R_{ij}	rule base
$\mathbf{R}_{LQ}$	real symmetric and positive definite matrix
t_f	final time
$u(t)$	control torque
u_F	FLC output
u_{FLC}	crisp control action from the FLC after denormalization
$v(x, t)$	deflection of a point located at a distance x
$V(t)$	Lyapunov function

The MS was received on 13 September 2002 and was accepted after revision for publication on 29 May 2003.
* *Corresponding author: Department of Automatic Control and Systems Engineering, University of Sheffield, Mappin Street, Sheffield S1 3JD, UK.*

W_{ji}, W_{kj}	connective weights between the input to hidden layers and output to hidden layers
x, x_s	spatial point and states
$y_d(t)$, $\dot{y}_d(t)$	desired tip displacement and velocity
$y_m(t)$	output of the reference model
$y_r(t)$	reflected tip position
$y_t(t)$, $\dot{y}_t(t)$	actual tip displacement and tip velocity of the flexible arm
α_{mo}, α_{mh}	momentum factors
η_k, η_j	learning parameters
θ	hub angle
λ_i	ith eigenvalue
$\mu_i(u)$	ith membership function for the output label
ρ	linear mass density of the link
$\phi_i(x)$	ith mode shape function
ω_i	frequency of the natural vibration of the ith mode
$\times$	Cartesian product
$\circ$	sup-min operation

1 INTRODUCTION

The commonest approach in the past for controlling flexible-link manipulators has been to design a controller based on an analytical system model. A comparative study of different flexible manipulator controllers such as PD (proportional-derivative), LQR (linear quadratic regulator), singular perturbation controller and feedback linearization controller has been made in reference [**1**]. However, the major cause of difficulty with model-based controllers is that their performance is crucially dependent on the accuracy of the manipulator model. As it is difficult to achieve an accurate model, performance therefore tends to be poor.

Fuzzy logic controllers (FLCs) offer an attractive alternative to conventional model-based control schemes. An FLC is basically a model-free control paradigm, where the control signal is calculated by fuzzy inference rather than from the system dynamics. This property makes an FLC suitable for controlling non-linear, uncertain or ill-understood dynamic systems such as flexible manipulator systems. It has also been proved that an FLC works well in situations where there is unknown variation in plant parameters and structures [**2**–**4**].

A number of investigations have reported on the application of fuzzy logic in rigid manipulator control ([**5**] and references therein). Recently, fuzzy logic methods have also been applied in flexible manipulators. Lin and Lee [**6**] proposed a PD–FLC for tip position control of a single-link flexible arm, using the integral square error as the performance index to tune the membership function to determine the optimal percentage of overlap. Unfortunately, the tuning scheme involved is tedious and time consuming. In another approach, Liu and Lewis [**7**] developed an FLC for a single-link flexible manipulator after feedback-linearizing the dynamic model, but this was only applied for rotor-angle tracking and did not provide tip-motion control. Unfortunately, tip-motion control is a much more difficult problem because of the unstable dynamics associated with the non-minimum phase property.

The significant contribution made by this current paper is that the PD–FLC scheme presented provides proper tip-motion control rather than just joint-angle control. The known tolerance of fuzzy controllers to moderate parameter variations means that the controller will perform well in many applications. However, some applications involve large payload changes and problems are to be expected in such cases if proper account is not taken of such changes.

The case where there are large payload changes has been considered by several authors. Moudgal *et al.* [**8**] proposed an indirect, adaptive, fuzzy model reference control that achieves faster slews with minimum end vibrations in situations involving unknown payload variations that give rise to changing plant dynamics. Mudi and Pal [**9**] proposed self-tuned fuzzy PI (proportional-integral) and PD controllers, where the output scale factors are adjusted on-line by a set of fuzzy rules based on the current trend of the controlled system. Several genetic algorithm (GA)-based fuzzy controller design methods have been proposed which determine the optimal controller parameters to achieve better FLC performance [**10**, **11**]. Most of these involve off-line determination of the controller parameters. Hence, the parameters so determined may not provide optimal FLC performance during actual operation of the robot.

Noting the deficiencies identified in these various attempts to implement an adaptive FLC, the work described in this paper proposes a novel neuro-fuzzy controller that considers the system parameter variations that are reflected through the reference model. The deviation between the model output and the plant output is used to train the neural network (NN) to adjust the output scale factor on-line. The tuning difficulties of the fuzzy model reference adaptive controller (FMRC) are thus addressed.

2 DESIGN OF THE FUZZY LOGIC CONTROLLER

A PD–FLC was designed and applied to control the tip position of the manipulator, since it is well known that a PD–FLC gives a faster transient response than a PI-type FLC. Figure 1 shows the PD–FLC structure for a single-link flexible robot.

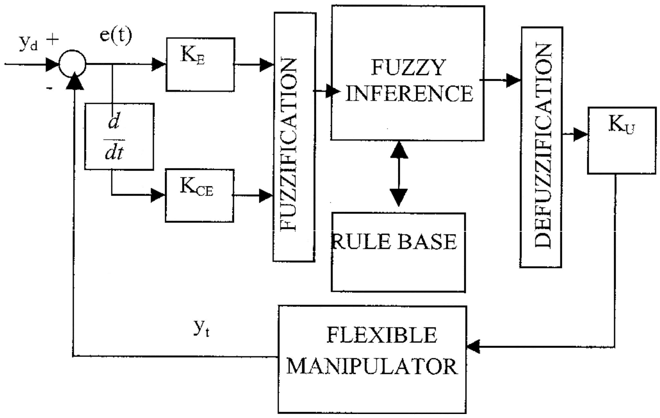

Fig. 1 Structure of the PD–FLC

The PD–FLC consists of a normalization unit, fuzzification interface, knowledge base, fuzzy inference system, defuzzification interface and a denormalization unit. In this design, the tip position error $e(t)$ and velocity error $\dot{e}(t)$, as defined below, are chosen as the input variables to the FLC, and the control torque $u(t)$ is considered as the output:

$$e(t) = y_d(t) - y_t(t) \tag{1}$$

$$\dot{e}(t) = \dot{y}_d(t) - \dot{y}_t(t) \tag{2}$$

where $y_t(t)$ and $\dot{y}_t(t)$ are the actual tip displacement and velocity of the flexible arm and $y_d(t)$ and $\dot{y}_d(t)$ are the desired tip displacement and velocity respectively. The input normalization block transforms the input variables of the FLC (e and $\dot{e}$) on the actual universe of discourse (UOD) (E and CE) to the normalized universe of discourse E_n and CE_n (e_n and ce_n) in the range of (−1.0 to 1.0), using the input scale factors K_E and K_{CE} for computational simplicity. The fuzzification block converts these crisp inputs to appropriate fuzzy sets using the membership functions as shown in Fig. 2. Here, seven symmetric triangular fuzzy sets, NB (negative big), NM (negative medium), NS (negative small), ZE (zero), PS (positive small), PM (positive medium) and PB (positive big), are used for both the input and output variables to the FLC.

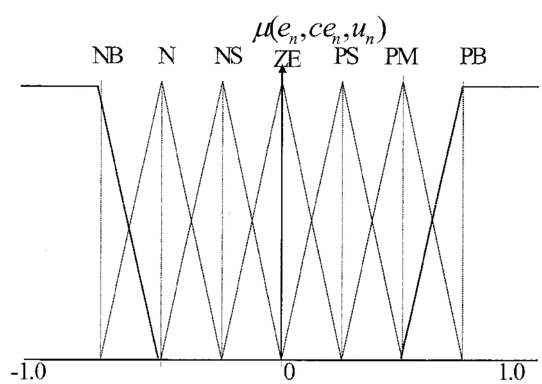

Fig. 2 Membership functions for FLC input and output variables

The knowledge base provides the membership functions and the linguistic control rules. The fuzzy inference engine performs fuzzy reasoning, based on the linguistic control rules, using Zadeh's compositional rule of inference [**12**]. The defuzzification block generates a crisp control output $u(t)$ by utilizing the centre of gravity method [**12**]:

$$u(t) = \frac{\sum_{i=1}^{n_r} \mu_i(u)\, u_i}{\sum_{i=1}^{n_r} \mu_i(u)} \tag{3}$$

where u_i is the centroid and $\mu_i(u)$ is the membership function value of the fuzzy set for the consequent (control action in this case) inferred at the ith quantization level on the control space (UOD); n_r is the number of quantization intervals. Just like input normalization, the output (u_n) in the computational UOD (Un) is denormalized by using the scale factor K_U to obtain the control action $u(t)$ in the actual UOD (U).

The first priority is to tune the scaling factors (SFs), because these are the global tuning parameters that affect the overall control performance. In adjusting these, consideration is given to rise time (t_r), overshoot (OS) and the steady state error. When the response is far away from the desired value, the input SFs are adjusted to reduce the rise time, and are later readjusted to prevent overshoot as the response approaches the desired value. The output SF is tuned to limit the FLC output to a reasonable value and to reduce the steady state error (e_{ss}). A basic manual tuning procedure that can be used for the FLC input and output SFs is given in Table 1.

Selection of appropriate fuzzy control rules is essential for obtaining efficient performance of the FLC. Several methods of deriving appropriate if–then fuzzy rules could be used [**8, 12**], but this paper uses an error response plane method [**6**]. The error response plane method is effectively a fuzzy logic control version of the well-known sliding mode controller. The error response plane shown in Fig. 3 is divided into three regions, namely I, II and III. Region III is the desired region of motion control, and the control torque should direct the position of the arm towards this region in the minimum possible time. Consider a point S1 in region I where the error signal is positive. In this case, there are three possibilities for the error slope $\dot{e}(t)$, i.e. positive, negative or zero. If the slope of the error signal is positive, there is a tendency for the system to move away from the desired region III. Therefore, to bring the system back to region III, a negative control signal needs to be applied. However, if the slope is negative, then the system may

Table 1 Tuning of scaling factors for the FLC

Increase in SF	Effect on t_r	Effect on OS	Effect on e_{ss}
K_E	Decrease	Increase	Decrease
K_{CE}	Increase	Decrease	Small change
K_U	Decrease	Increase	Decrease

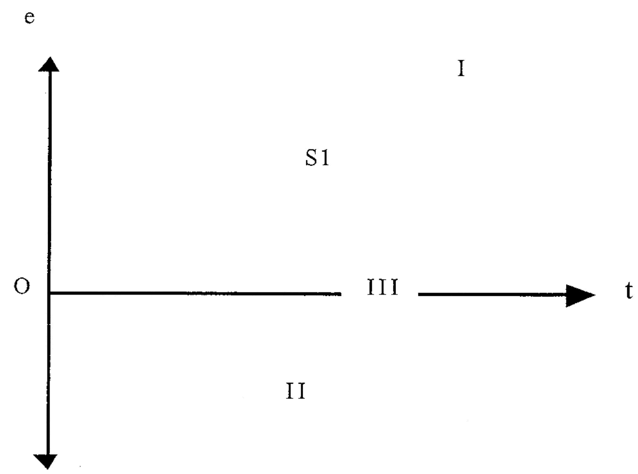

Fig. 3 Error response method for deriving fuzzy rules

have an inclination to produce overshoot from the desired region, suggesting that a positive control torque should be applied. Finally, if the error slope is zero, this implies that there may be a steady state error. Hence, in order to bring the system back to region III, a positive torque should be applied so that the system has a positive error slope at the next iteration. By doing so, the negative control signal is activated and drives the system to the desired region. The rules pertaining to regions II and III can be obtained using similar reasoning. In this way 49 rules were constructed, as shown in Table 2.

Using the compositional rule of inference, the fuzzy control is

$$U = (E \times CE) \circ R \tag{4}$$

where R is the rule base, $\times$ is the Cartesian product and $\circ$ is the sup-min operation. It may be noted here that the antecedent indices j and k respectively for $E(e_j)$ and $CE(\dot{e}_k)$ in the rule base are used to access the corresponding consequent $U(u_i)$ for the ith rule, R_i (which is the same as R_{jk}). The control action $U(u_i)$ for the ith rule can be obtained using

$$U(u_i) = \{E(e_j) \cap CE(\dot{e}_k) \cap R_{jk}(e_j, \dot{e}_k, u_i)\} \tag{5}$$

and a maximum operation is performed over all the rules

Table 2 Fuzzy rule base for the PD–FLC

U			$E\rightarrow$				
CE ↓	NB	NM	NS	ZE	PS	PM	PB
NB	PS	PS	PS	NB	NM	NM	NB
NM	PM	PS	PS	NM	NS	NM	NB
NS	PB	PM	PS	NS	NS	NM	NM
ZE	PM	PS	PS	ZE	NS	NS	NM
PS	PB	PM	PS	PM	NS	NS	NB
PM	PB	PM	PS	PM	NS	NS	NM
PB	PB	PM	PM	PB	NS	NS	NS

to find the fuzzy control vector as

$$U(u) = \bigcup_{i=1,\ldots,n} \{E(e_j) \cap CE(\dot{e}_k) \cap R_{jk}(e_j, \dot{e}_k, u_i)\}$$

where n ($n = jl$) is the number of rules.

3 FUZZY MODEL REFERENCE ADAPTIVE CONTROLLER

As discussed in the Introduction, section 1, the fuzzy controllers discussed in section 2 will work satisfactorily provided that the manipulator system dynamics are relatively undisturbed. However, when the dynamics of the robot are varied by a large payload change, the FLC parameters need re-tuning to maintain good control performance. An alternative to the tedium of parameter re-tuning is to design an adaptive FLC, where the on-line re-tuning algorithm for the FLC parameters is based on the reference model. One such scheme is the FMRC [**8**] shown in Fig. 4, which consists of four main blocks, i.e. the system to be controlled (FM), the conventional FLC to be tuned, a reference model (REF MODEL), which carries the performance objective information, and a learning mechanism. The FMRC only tunes the output membership functions and does not affect the input membership functions. The learning mechanism tunes the rule base of the direct fuzzy controller so that the closed system behaves like the reference model. The learning mechanism consists of two parts, namely a fuzzy inverse model and a knowledge base modifier. The fuzzy inverse model maps $e_r(t)$ to changes in the system inputs $\gamma(t)$ that are necessary to force $e_r(t)$ to zero; $e_r(t)$ is obtained by comparing the actual tip position with the output of the reference model $y_m(t)$: $e_r(t) = y_m(t) - y_t(t)$. The knowledge base modifier adjusts the rule base of the FLC to effect the changes needed in the control torque. K_{EP}, K_{CEP} and K_{UP} are the scaling factors of the fuzzy inverse model, which are similar to the

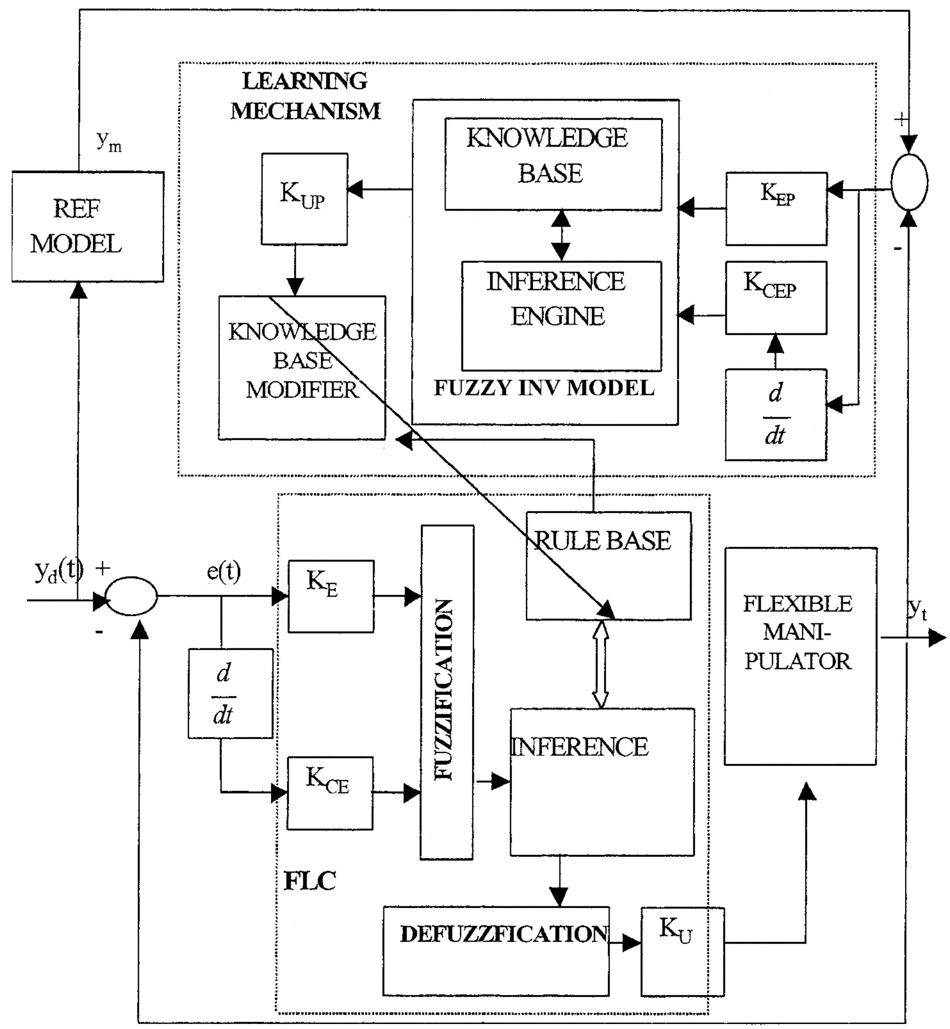

Fig. 4 Fuzzy model reference adaptive controller

scaling parameters of the FLC (K_E, K_{CE} and K_U), as described earlier.

The knowledge base modifier changes the rule base of the FLC so that the previously applied control action is modified by an amount $\gamma(k)$, where k is the iteration number. Consider the previously computed control action $u(k-1)$ and assume that it contributed to the present good or bad system performance, i.e. such that the value of $y_t(k)$ does or does not match the model output $y_m(k)$. Now, with the error and change of error as $e(k-1)$ and $ce(k-1)$, the rule base of the FLC can be modified to produce a desired output:

$$u(k-1)+\gamma(k) \tag{6}$$

Let $c_i(k)$ be the centre of the ith output membership function at iteration k. For all rules in the active set, modification of the output membership function centres can be achieved using the following relation:

$$c_i(k)=c_i(k-1)+\gamma(k) \tag{7}$$

It may be noted that the rules that are not in the active set do not have their output membership functions modified.

4 NEURO-FUZZY CONTROLLER

It is well known that, although FLCs work well with imprecise dynamics or even with no knowledge about the system dynamics, they do not have a learning capability of their own. However, a learning mechanism is created if NNs, which have good learning attributes, are hybridized with fuzzy systems. The resulting systems are popularly known as fuzzy neural systems or neuro-fuzzy systems. Various neuro-fuzzy systems have been reported that use NNs to modify FLC parameters such as scaling factors, membership functions and the rule base [**13–16**]. Neuro-fuzzy systems can usually be represented as multilayered feedforward networks, such as ANFIS [**15**], FuNe [**17**] and NEFCON [**18**]. Sometimes a four-layer architecture is used [**14**], where the membership functions are represented in the neurons of the second layer and the third layer is a rule layer followed by a fourth layer for the FLC output. A neuro-fuzzy system for on-line tuning of the output scale factors was proposed by Lin *et al.* [**16**], but this has no mechanism for adjusting the input scaling factors. In addition, this FLC was designed with quantized input and output variables, which may not be a good choice for controlling physical systems like flexible manipulators that have very complex dynamics.

In contrast, the work presented in this paper proposes a hybrid neuro-fuzzy controller that tunes both the input and output scale factors of the FLC by means of a three-layered perceptron neural network. Also, continuous UODs for the FLC input and output variables are used in the FLC design instead of a discrete UOD. The proposed scheme (Fig. 5) incorporates FLC, NN and PD controller blocks. The purpose of using the PD controller is to enhance the rise time of the system output during the initial learning phase of the NN. A multilayer perceptron neural network (MLPNN) is used to build the hybrid neuro-fuzzy controller applied to the flexible manipulator. The MLPNN can have many layers with a reasonable number of nodes in each one.

A three-layer NN is employed for on-line tuning of the input scale factors (K_E and K_{CE}) and the output scale factor (K_U) of the FLC, where the NN inputs are selected as $e(t) = y_t(t) - y_d(t)$ and $\dot{e}(t) = \dot{y}_t(t) - \dot{y}_d(t)$. The signals to the input layer are not weighted and are therefore given as $net_i = x_i$, where x_i represents the ith input to the node of the input layer. The output of the ith neuron in this layer is $Q_i = f_i(net_i) = net_i$. For the hidden layer, the signal input and the output of the jth neuron can be expressed as

$$net_j = \sum_{i}^{n_i} (W_{ji} O_i) + b_j, \qquad O_j = f_j(net_j) = \frac{1}{1 + e^{-net_j}} \tag{8}$$

where W_{ji} are the connection weights between the input and the hidden layer, b_j are the threshold values for the units in the hidden layer, n_i is the number of nodes in the input layer and f_j is the sigmoidal activation

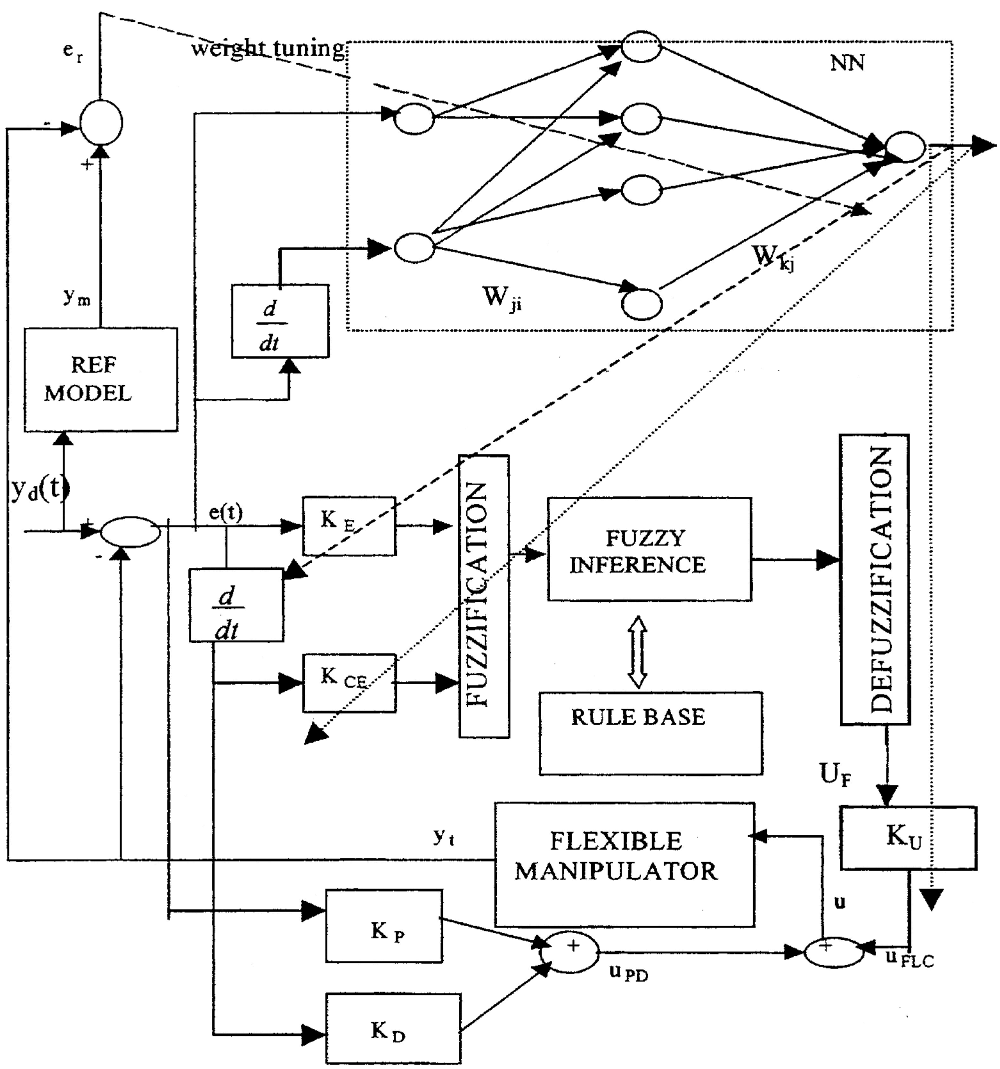

Fig. 5 Hybrid neuro-fuzzy controller

function. Finally, the signal and activation for the output layer of the NN are given by

$$net_k = \sum_j^{n_\mathrm{h}} (W_{kj} O_j) + b_k, \qquad O_k = f_k(net_k) \tag{9}$$

where W_{kj} are the connection weights between the output and the hidden layer, b_k are the threshold values for the units in the output layer, n_h is the number nodes in the hidden layer and f_k is the sigmoidal activation function.

4.1 Training of the NN

The on-line training algorithm for the NN can be derived in terms of the error function E_N as

$$E_\mathrm{N} = \tfrac{1}{2} e_\mathrm{r}^2 = \tfrac{1}{2}(y_\mathrm{m} - y_\mathrm{t})^2 \tag{10}$$

Then, the learning algorithm is as follows.

4.1.1 *Output layer*

The weights are updated using the steepest descent method:

$$\Delta W_{kj} = -\eta_k \frac{\partial E_\mathrm{N}}{\partial W_{kj}} = -\eta_k \frac{\partial E_\mathrm{N}}{\partial net_k} \frac{\partial net_k}{\partial W_{kj}} = \eta_k \delta_k O_j \tag{11}$$

where the factor η_k is the learning parameter for the connection weights between the output and the hidden layers. The weights of the output layer are updated according to the back-propagation algorithm. In order to increase the learning rate without leading to oscillation in the output response, momentum factors α_mo and α_mh are included in the adapting weights W_{kj} and W_{ji} [**19**]:

$$W_{kj}(t+1) = W_{kj}(t) + \Delta W_{kj}(t) + \alpha_\mathrm{mo} \Delta W_{kj}(t) \tag{12}$$

The error term to be propagated is given by

$$\delta_k = -\frac{\partial E_\mathrm{N}}{\partial net_k} = -\frac{\partial E_\mathrm{N}}{\partial e_\mathrm{r}} \frac{\partial e_\mathrm{r}}{\partial y_\mathrm{t}} \frac{\partial y_\mathrm{t}}{\partial u_\mathrm{FLC}} \frac{\partial u_\mathrm{FLC}}{\partial O_k} \frac{\partial O_k}{net_k} \tag{13}$$

where $u_\mathrm{FLC} = u_\mathrm{F} K_U = u_\mathrm{F} O_k$, u_FLC is the crisp control action from the FLC after denormalization, u_F is the FLC output and O_k is the kth node in the output layer. The output layer consists of three nodes, as shown in Fig. 5, corresponding to the scale factors (K_E, K_{CE}, K_U). The errors propagated to these nodes for $k = 1, 2, 3$ are as follows.

Error for k = 1 (output scale factor)

$$\delta_k = -\frac{\partial E_\mathrm{N}}{\partial e_\mathrm{r}} \frac{\partial e_\mathrm{r}}{\partial y_\mathrm{t}} \frac{\partial y_\mathrm{t}}{\partial u_\mathrm{FLC}} \frac{\partial u_\mathrm{FLC}}{\partial O_k} \frac{\partial O_k}{net_k} \tag{14}$$

The Jacobian of the system $\partial y_\mathrm{t} / \partial u_\mathrm{FLC}$ can be approximated as

$$\frac{\partial y_\mathrm{t}}{\partial u_\mathrm{FLC}} = \begin{cases} M, & \dfrac{\partial y_\mathrm{t}}{\partial u_\mathrm{FLC}} > 0 \\ -M, & \dfrac{\partial y_\mathrm{t}}{\partial u_\mathrm{FLC}} < 0 \end{cases}$$

where M is the known bound of the manipulator system, which can be considered as a finite slew rate. Therefore, equation (14) can be modified to

$$\delta_k = e_\mathrm{r}(\pm M) u_\mathrm{F} \frac{\partial u_\mathrm{FLC}}{\partial net_k} = e_\mathrm{r}(\pm M) u_\mathrm{F} f'(net_k) \tag{15}$$

Error for k = 2 and 3 (input scale factors)

$$\delta_k = -\frac{\partial E_\mathrm{N}}{\partial e_\mathrm{r}} \frac{\partial e_\mathrm{r}}{\partial y_\mathrm{t}} \frac{\partial y_\mathrm{t}}{\partial u_\mathrm{FLC}} \frac{\partial u_\mathrm{FLC}}{\partial O_k} \frac{\partial O_k}{net_k} = e_\mathrm{r} \left(\frac{\partial y_\mathrm{t}}{\partial O_k} \right) \frac{\partial O_k}{\partial net_k} \tag{16}$$

To simplify computation, $\partial y_\mathrm{t} / \partial O_k$ can be approximated by a bound N similar to the one used for approximating $\partial y_\mathrm{t} / \partial u_\mathrm{FLC}$. Therefore, equation (16) becomes

$$\delta_k = e_\mathrm{r}(\pm N) f'(net_k) \tag{17}$$

4.1.2 *Hidden layer*

The error term to be propagated is given by

$$\delta_j = -\frac{\partial E_\mathrm{N}}{\partial net_j} = -\frac{\partial E_\mathrm{N}}{\partial net_k} \frac{\partial net_k}{\partial O_j} \frac{\partial O_j}{net_j} \tag{18}$$

with weights updated according to

$$\Delta W_{ji} = -\eta_j \frac{\partial E_\mathrm{N}}{\partial W_{ji}} = -\eta_j \frac{\partial E_\mathrm{N}}{\partial net_j} \frac{\partial net_j}{\partial W_{ji}} = \eta_j \delta_j O_i \tag{19}$$

where the factor η_j is the learning parameter for adapting the connection weights between the hidden and the input layers. The weights of the hidden layer are updated according to

$$W_{ji}(t+1) = W_{ji}(t) + \Delta W_{ji}(t) + \alpha_\mathrm{mh} W_{ji}(t) \tag{20}$$

The bias of each neuron in the hidden and output layers is trained on-line using the same learning rate parameters.

4.2 Stability of the NFC

By choosing suitable values for the learning parameters of the connection weights between the hidden and input layers (η_j) and the output and hidden layers (η_k), the convergence of the NFC is guaranteed. This is shown as follows.

Lemma 1 [**14**]

If $f(a) = a - a^2$, then $f(a) \leqslant 0.25, \forall a \in [0, 1]$.

Theorem (convergence of NFC)

If η_j and η_k are chosen as

$$\eta_k = \frac{1}{(L_{kj\max} u_F)^2} = \frac{16}{P_{kj} u_F^2)}$$

and

$$\eta_j = \frac{1}{(L_{ji\max} u_F)^2} = \frac{256}{|W_{kj}|^2_{\max} P_{ji} u_F^2}$$

then the convergence of the NFC is guaranteed, where $L_{kj\max}$ and $L_{ji\max}$ are defined as

$$L_{kj\max} = \max_t \| L_{kj}(t) \|$$

and

$$L_{ji\max} = \max_t \| L_{ji}(t) \|$$

with

$$L_{kj}(t) = \frac{\partial O_k}{\partial W_{kj}}$$

$$L_{ji}(t) = \frac{\partial O_k}{\partial W_{ji}}$$

$\| \cdot \|$ is the Euclidean norm in $\Re^n$ and $W_{kj\max}$ is defined as

$$W_{kj\max} = \max_t \| W_{kj}(t) \|$$

P_{kn} is the number of weights between the output and hidden layer in the NN and P_{jn} is the number of weights between the hidden and output layers.

Proof. For a sigmoidal activation function

$$f_k'(net_k) = f_k(net_k)[1 - f_k(net_k)]$$

Using Lemma 1,

$$f_k'(net_k) = f_k(net_k)[1 - f_k(net_k)] \leqslant 0.25 \qquad \text{for } f_k(net_k) \in [0, 1]$$

$L_{kj}(t)$ can be written as

$$L_{kj}(t) = \frac{\partial O_k}{\partial W_{kj}} = \frac{\partial O_k}{\partial net_k} \frac{\partial net_k}{\partial W_{kj}} = f_k'(net_k) O_j \leqslant 0.25 O_j \tag{21}$$

Therefore, from equation (21),

$$\| L_k(t) \| \leqslant \sum_{n=1}^{j} \frac{\sqrt{P_{kn}}}{4} \tag{22}$$

Let $V(t)$ be a Lyapunov function chosen as

$$V(t) = \tfrac{1}{2} e_r^2(t) \tag{23}$$

where $e_r(t)$ is the tracking error. From equation (23), the change in Lyapunov function is

$$\Delta V = \frac{V(t+1) - V(t)}{2h} = \frac{e_r^2(t+1) - e_r^2(t)}{2h} \tag{24}$$

where h denotes the step size. The tracking error $e_r(t+1)$ is

$$e_r(t+1) = e_r(t) + \left[\frac{\partial e_r(t)}{\partial W_k} \right]^T \Delta W_k \tag{25}$$

where ΔW_k denotes the change in weights of the NN between the hidden and the output layers. Replacing the Jacobian of the system by its sign function using equation (12) gives

$$\begin{aligned} \| e_r(t+1) \| &= \| e_r(t) [1 - \eta_k u_F^2 L_{kj}^T L_{kj}] \| \\ &\leqslant \| e_r(t) \| \, \| [1 - \eta_k u_F^2 L_{kj}^T L_{kj}] \| \end{aligned} \tag{26}$$

If η_k is chosen as

$$\eta_k = \frac{1}{(L_{kj\max} u_F)^2} = \frac{16}{P_{kj} u_F^2}$$

then the term $\| [1 - \eta_k u_F^2 L_{kj}^T L_{kj}] \|$ in equation (26) is less than 1. Similarly, $L_{ji}(t)$ can be written as

$$\begin{aligned} L_{ji}(t) &= \frac{\partial O_k}{\partial W_{ji}} = \frac{\partial O_k}{\partial net_k} \frac{\partial net_k}{\partial O_j} \frac{\partial O_j}{\partial net_j} \frac{\partial net_j}{\partial W_{ji}} \\ &= f_k'(net_k) \sum_j W_{kj} f_j' \sum_i O_i \end{aligned} \tag{27}$$

Now combining the bounds of $f_k'(\cdot)$ and $f_j'(\cdot)$, equation (27) can be written as

$$L_{ji}(t) \leqslant \frac{1}{16} |W_{kj}|_{\max} |O_i|_{\max} = \frac{\| W_{kj} \| \, \| O_i \|}{16} \tag{28}$$

Hence

$$\| L_j(t) \| \leqslant \sum_{n=1}^{i} \sqrt{\frac{P_{ji}}{4}}$$

The change in error can also be written in a similar fashion to equation (25) in terms of W_j and weight changes ΔW_j (a vector of weight changes from the hidden layer to the input layer).

Because the Jacobian of the system is replaced by its sign function, equation (19) can be used to give

$$\begin{aligned} \| e_r(t+1) \| &= \| e_r(t) [1 - \eta_j u_F^2 L_{ji}^T L_{ji}] \| \\ &\leqslant \| e_r(t) \| \, \| [1 - \eta_j u_F^2 L_{ji}^T L_{ji}] \| \end{aligned} \tag{29}$$

From equation (29), it can be seen that, if η_j is chosen as

$$\eta_j = \frac{1}{(L_{ji\max} u_F)^2} = \frac{256}{|W_{kj}|^2_{\max} P_{ji} u_F^2}$$

then $\| [1 - \eta_j u_F^2 L_{ji}^T L_{ji}] \| < 1$. Thus, the Lyapunov stability ($V > 0$ and $\Delta V < 0$) is guaranteed. The tracking error $e_r(t) \to 0$ as $t \to \infty$. Therefore, the theorem has been proved.

5 RESULTS AND DISCUSSION

5.1 Performance of the FLC compared with the PD and LQR

In order to demonstrate the superior performance of the FLC, an analytic system model was developed and two alternative model-based controllers based on the PD and LQR approaches were designed. It has already been noted that the performance of model-based controllers is crucially dependent on the accuracy of the dynamic system model. Many previously published models have inaccuracies, but recent work [**20**] has described a model of improved accuracy: this forms the basis of the model used for this current work. Two model-based controllers were developed: firstly, a proportional-derivative controller (PDC) and, secondly, a linear quadratic regulator (LQR).

Figures 6 to 9 compare the results obtained with the PD, FLC and LQR for tip position control when the flexible manipulator was commanded to move from an initial position of 0 rad to a target tip position of 0.5 rad, with the parameters of the FLC set at $K_E = 0.4$, $K_{CE} = 0.15$, $K_U = 5.0$. The first mode trajectories with the PD, FLC and LQR are compared in Fig. 6. The first mode of vibration is damped faster and has a smaller amplitude with the FLC compared to the other two controllers. (Although not shown, to save space, the FLC also has the smallest second modal vibration and damps it in the least time.) From the tip deflection trajectories shown in Fig. 7, it can be seen that deflection is less with the LQR than for the PD and FLC. However, the FLC damps out the deflection faster compared to the other controllers. Figure 8 shows the tip position trajectories for the PD, FLC and LQR. The PD controller gives more overshoot compared with the FLC and LQR. The tip position trajectory with the PD has a fast rise time but overshoots more than the FLC. The tip position with the LQR has a delayed rise time and higher overshoot compared with the FLC. Control profiles of the controllers are shown in Fig. 9. Initially, the control torque rises to a maximum of 2.5, 2.35 and 2.3 N m with the PD, LQR and FLC respectively, and in all cases the control torque eventually becomes zero when the desired tip displacement is achieved and the vibration is completely damped out.

5.2 Performance of the NFC compared with the FMR

Next, the effectiveness of the proposed NFC is compared with the FMRC. For the FMRC and the proposed NFC, the reference model is taken from reference [**8**], $G(s) = K_r/(s + a_r)$, where $a_r = 3.0$ and $K_r = 3.0$. The model is discretized at the same sampling time of 0.001 s. The structure of the NN for the NFC was chosen with two input nodes, 20 hidden nodes and two output nodes. The two inputs to the NN, as discussed in section 4, are the tip position error $e(t)$ and the tip velocity error $\dot{e}(t)$. It has been confirmed through different trial runs of the NFC that choosing 20 hidden neurons gives the best results. The initial weights were set with small random values in the range of ± 0.1. The momentum factors α_{mo} and α_{mh} were chosen as 0.1 and 0.15 respectively.

Figure 10 gives the tip position trajectories obtained with the NFC and FMRC, showing that the NFC performs better than the FMRC. Figure 11 compares the first mode trajectories and shows that the amplitude of this modal vibration is less for the NFC than the FMRC. (Although not shown, to save space, the second mode

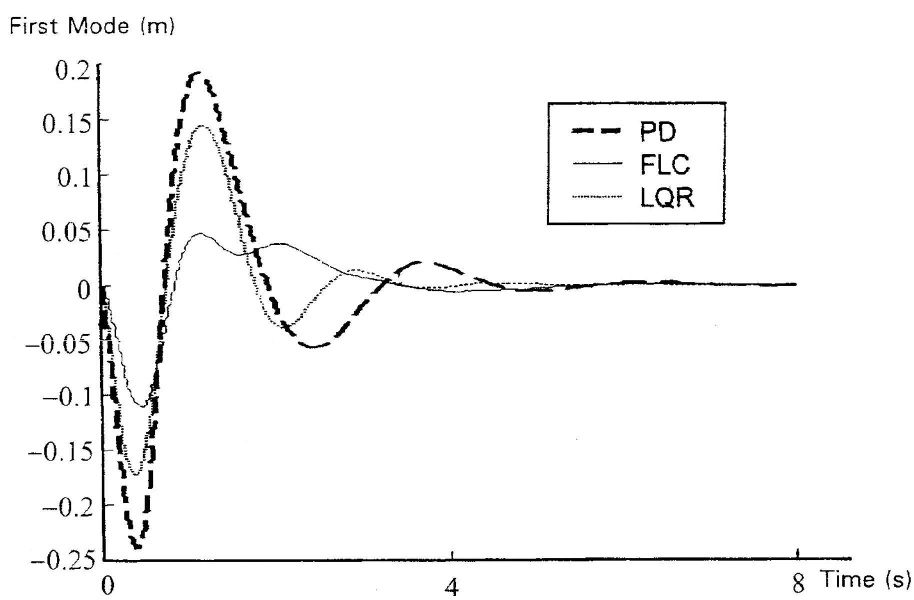

Fig. 6 Comparison of first mode trajectories with the PD, LQR and FLC

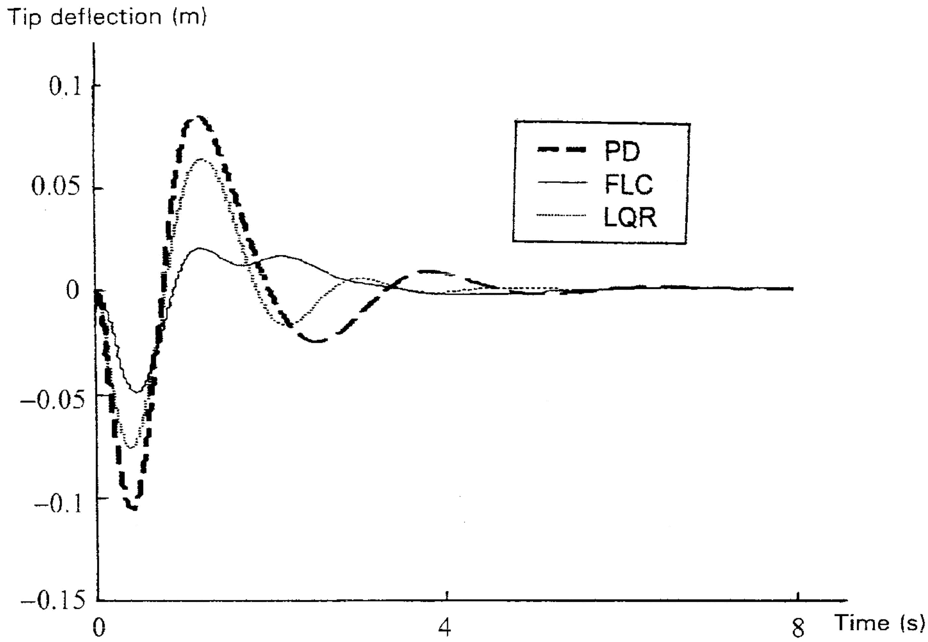

Fig. 7 Comparison of tip deflection trajectories with the PD, LQR and FLC

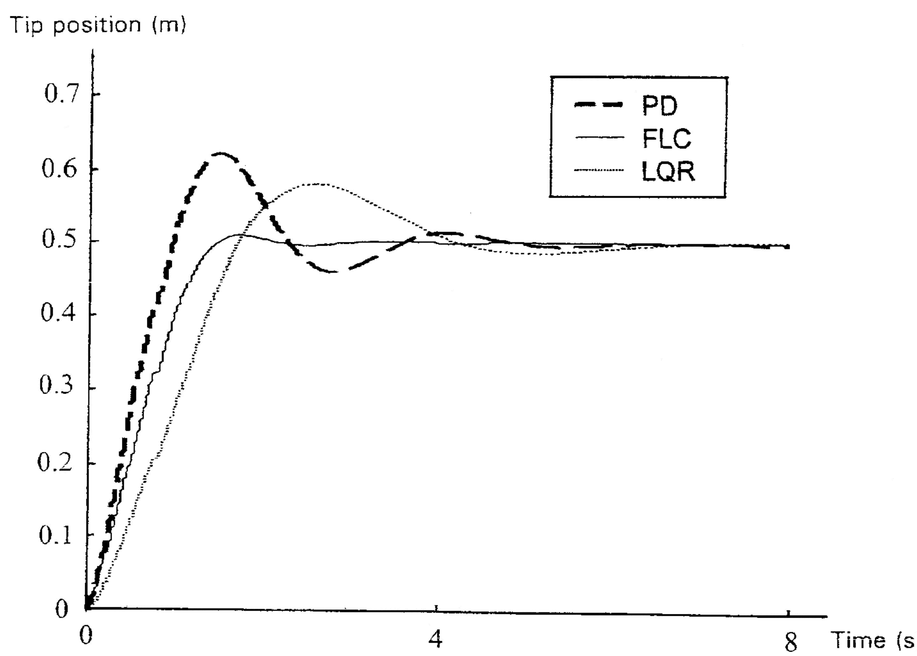

Fig. 8 Tip position trajectories with the PD, LQR and FLC

of vibration also has a smaller amplitude with the NFC.) Figure 12 compares the tip vibrations and it is obvious that the NFC damps out the end vibration more effectively. The control signals generated are compared in Fig. 13. Initial torques of 10.2 and 9.8 N m respectively were produced with the NFC and FMRC at maximum deflection, but then the control torques decay to zero as the tip position error reduces to zero.

6 CONCLUSIONS

As explained in section 1, model-based controllers such as the PD and LQR generally perform poorly owing to inaccuracies in the models on which they depend. Fuzzy logic controllers, because they do not require *a priori* development of an analytic system model, can potentially perform much better. Previous applications of the

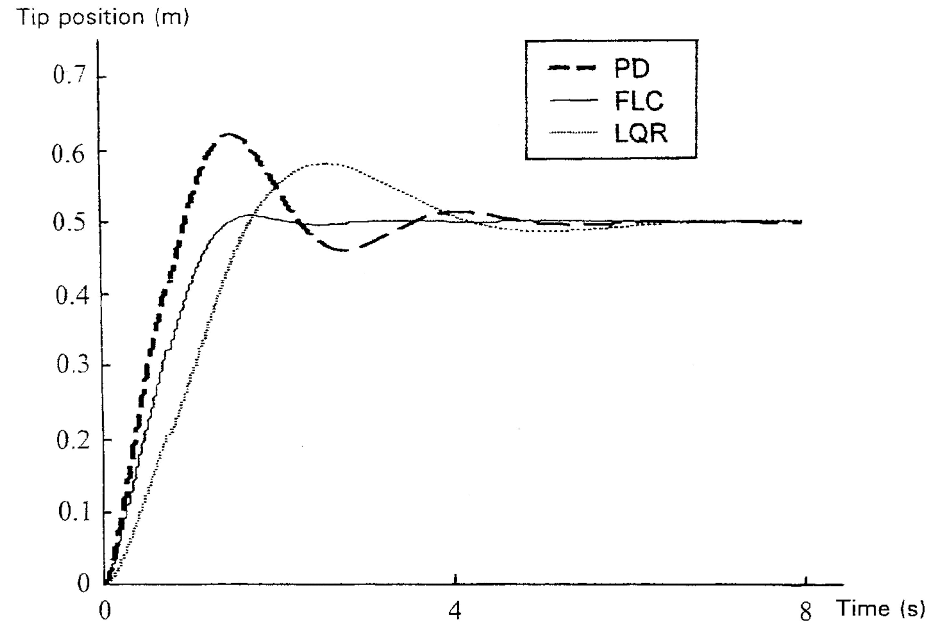

Fig. 9 Control torque requirements with the PD, LQR and FLC

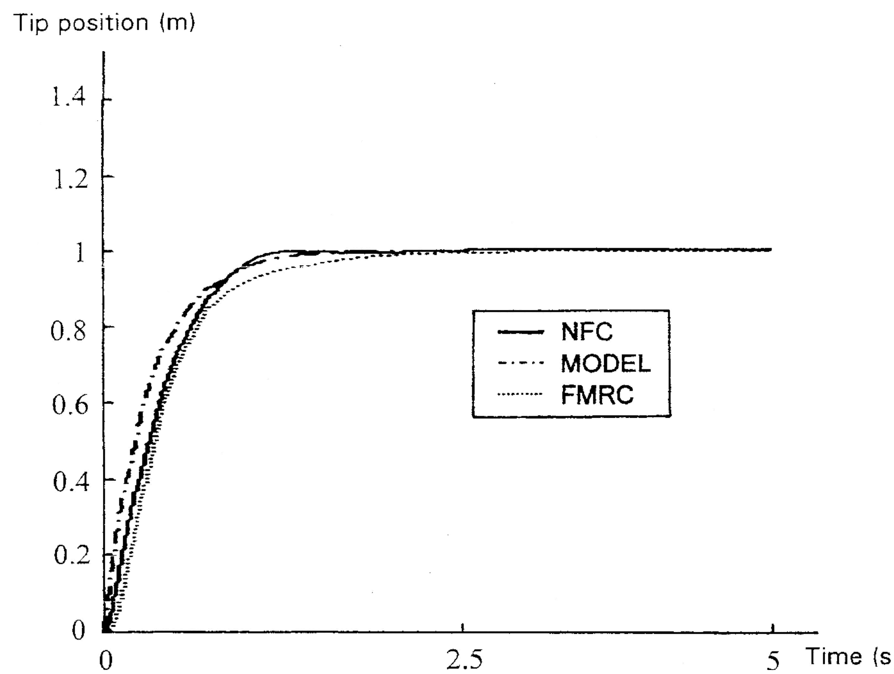

Fig. 10 Comparison of tip regulation performance with the NFC and FMRC

FLC to flexible manipulators have been deficient in various respects as discussed, and the contribution made by this paper has been in the development of an FLC that provides good control of tip motion in the manipulator. Simulation results have confirmed the superior performance compared with model-based control schemes.

In the case where there is significant payload variation, the necessary re-tuning of the FLC parameters is tedious. Previous work has proposed a fuzzy model reference adaptive controller (FMRC) that tunes the rule base of the FLC and the membership functions on-line. However, this approach is not always successful because it sometimes becomes difficult for the FMRC to tune six scale parameters (K_E, E_{CE} and K_U for the FLC and K_{EP}, K_{CEP} and K_{UP} for the fuzzy inverse model). To avoid this problem, a neuro-fuzzy controller has been described that uses an NN to tune the input and the output scale factor parameters of the FLC on-line. The

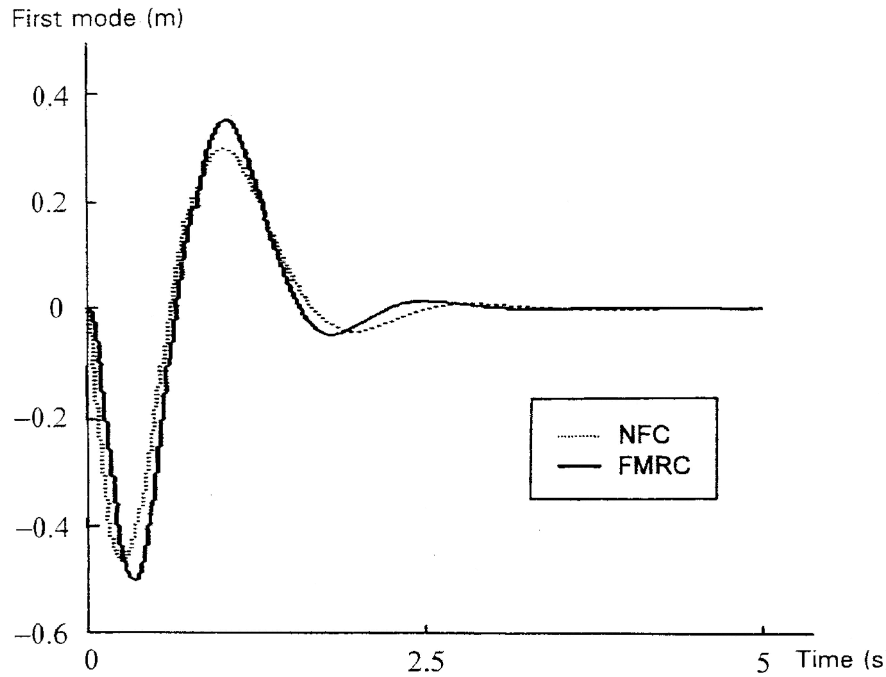

Fig. 11 First mode suppression performance with the NFC and FMRC

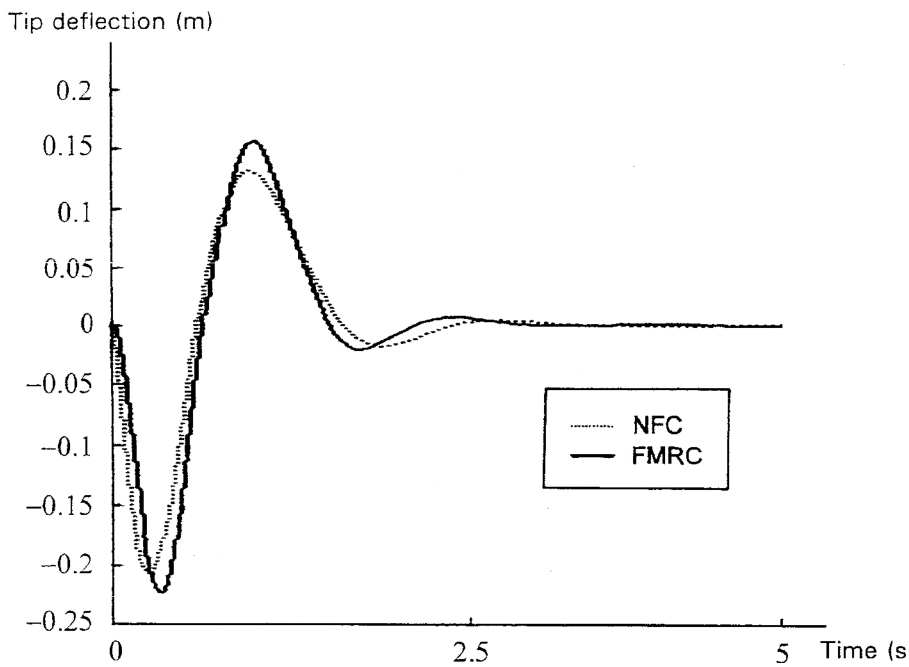

Fig. 12 Tip deflection curves with the NFC and FMRC

results presented have compared the performance of this with that of an alternative fuzzy adaptive controller reported previously [**8**]. This comparison has shown the superior performance of the neuro-fuzzy adaptive controller developed. A particular advantage of the new controller is that it does not require knowledge of a mathematical model. It has also been shown that the performance of the FLC can be enhanced by the adaptive scale factors tuning algorithm. The proposed hybrid neuro-fuzzy controller provides a fast response when applied on-line to the flexible manipulator system by utilizing the good transient state performance of the PD controller. As the proposed controller tunes the parameters on-line, unlike the off-line GA-based optimized FLC, it is more likely to be more suitable for real-time applications.

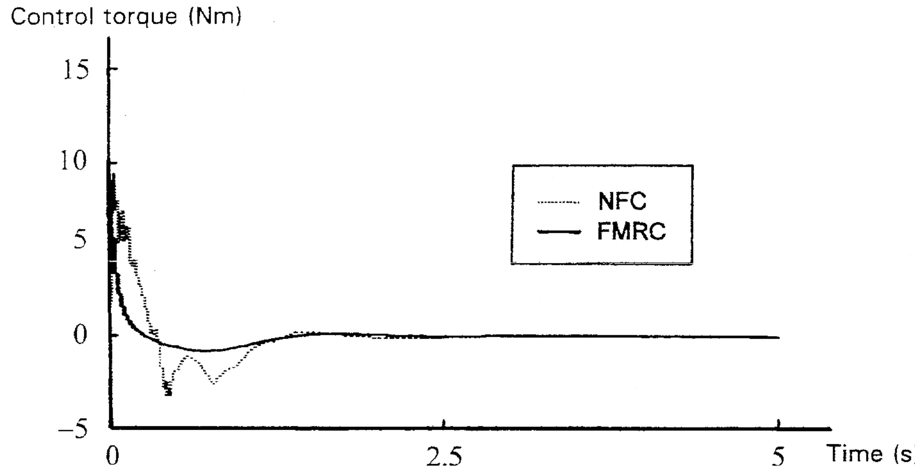

Fig. 13 Control torque profiles with the NFC and FMRC

REFERENCES

1 **Aoustin, Y., Chevallereau, C., Glumineeau, A.** and **Moog, C. H.** Experimental results for the end-effector control of a single flexible robot arm. *IEEE Trans. Control Systems Technol.*, 1994, **2**(4), 371–381.

2 **Mamdani, E. H.** and **Assilian, S.** An experiment in linguistic network with a fuzzy logic controller. *Int. J. Man–Machine Studies*, 1975, **7**(1), 1–13.

3 **Sayyarrodsari, B.** and **Homaifar, A.** The role of hierarchy in the design of fuzzy logic controllers. *IEEE Trans. Systems, Man, and Cybernetics, Part B: Cybernetics*, 1997, **27**(1), 108–118.

4 **Suh, I. H., Eom, K. S., Yeo, H. J.** and **Oh, S. R.** Fuzzy adaptive control of industrial robot manipulators with position servos. *Mechatronics*, 1995, **5**(8), 899–918.

5 **Soo, Y. Y.** and **Chung, J. M.** A robust fuzzy logic controller for manipulators with uncertainties. *IEEE Trans. Systems, Man, and Cybernetics, Part B: Cybernetics*, 1997, **27**(4), 707–713.

6 **Lin, Y. J.** and **Lee, T. S.** An investigation of fuzzy logic control of flexible robots. *Robotica*,1993, **11**(3), 363–371.

7 **Liu, K.** and **Lewis, F. L.** Hybrid feedback linearization/fuzzy logic control of a flexible link manipulator. *J. Intell. Fuzzy Systems*, 1994, **2**, 325–336.

8 **Moudgal, V. G., Passino, K. M.** and **Yurkovitch, S.** Rule-based control for flexible-link robot. *IEEE Trans. Control Systems Technol.*, 1994, **12**, 393–405.

9 **Mudi, R. K.** and **Pal, N. R.** A robust self-tuning scheme for PI and PD type fuzzy controllers. *IEEE Trans. Fuzzy Systems*, 1999, **7**(1), 3–16.

10 **Karr, C. L.** and **Gentry, E. J.** Fuzzy control of PH using genetic algorithms. *IEEE Trans. Fuzzy Systems*, 1993, **1**(1), 46–53.

11 **Zhou, Y. S.** and **Lai, L.-Y.** Optimal design for fuzzy controllers by the genetic algorithms. *IEEE Trans. Industry Applic.*, 2000, **36**(1), 93–97.

12 **Kotnik, P. T., Yurkovitch, S.** and **Özgüner, Ü.** Acceleration feedback control of a flexible manipulator. *J. Robotic Systems*, 1988, **5**(3), 181–196.

13 **Chen, M.** and **Linkens, D. A.** A hybrid neuro-fuzzy controller. *Fuzzy Sets and Systems*, 1998, **99**, 27–36.

14 **Halgamauge, S. K.** and **Glesner, M.** Neural networks for designing fuzzy systems for real-world applications. *Fuzzy Sets and Systems*, 1994, **65**, 1–2.

15 **Jang, J.-S. R.** ANFIS: adaptive network based fuzzy inference systems. *IEEE Trans. Systems, Man, and Cybernetics*, 1993, **23**(3), 665–685.

16 **Lin, F. J., Jong, W. R.** and **Wang, S. L.** A fuzzy neural network controller for parallel-resonant ultrasonic motor drive. *IEEE Trans. Ind. Electronics*, 1998, **45**(6), 929–937.

17 **Nauck, D., Klawonn, F.** and **Kruse, R.** *Foundations of Neuro-Fuzzy Systems*, 1996 (John Wiley, Chichester).

18 **Lin, F.-J.** and **Wai, R.-J.** A hybrid computed torque controller using fuzzy neural network for motor-quick-return servo mechanism. *IEEE Trans. Mechatronics*, 2001, **6**(1), 85–89.

19 **Omatu, S., Khalid, M.** and **Yusof, R.** *Neuro-Control and Its Applications*, 1996 (Springer, London).

20 **Morris, A. S.** and **Madani, A.** Static and dynamic modelling of a two-flexible-link manipulator. *Robotica*, 1996, **14**, 289–300.