Robotica (2002) volume 20, pp. 417–427. © 2002 Cambridge University Press
DOI: 10.1017/S026357470200406X Printed in the United Kingdom

Tracking control of a flexible-link manipulator using neural networks: experimental results

H. A. Talebi*, K. Khorasani** and R. V. Patel†

(Received in Final Form: January 22, 2002)

SUMMARY

In this paper, the problem of tip position tracking control of a flexible-link manipulator is considered. Two neural network schemes are presented. In the first scheme, the controller is composed of a stabilizing joint PD controller and a neural network tracking controller. The objective is to simultaneously achieve hub-position tracking and control of the elastic deflections at the tip. In the second scheme, tracking control of a point along the arm is considered to avoid difficulties associated with the output feedback control of a non-minimurn phase flexible manipulator. A separate neural network is employed for determining an appropriate output to be used for feedback. The controller is also composed of a neural network tracking controller and a stabilizing joint PD controller. Experimental results on a single-link flexible manipulator show that the proposed networks result in significant improvements in the system response with an increase in controller dynamic range despite changes in the desired trajectory.

KEYWORDS: Flexible-link manipulators; Neural networks; Tracking control; Non-minimum phase systems, Output re-definition

1. INTRODUCTION

The problem of modeling and control of flexible-link manipulators has received much attention in the past several years. There are a number of potential advantages that arise from the use of light-weight flexible-link manipulators. For instance, in designing a space manipulator, energy efficiency must be considered. The use of light-weight manipulators also results in higher speed manipulations and a higher ratio of payload to arm weight compared to conventional rigid manipulators.

However, while taking advantage of the light-weight structure, one should also be concerned with the inherent complexities involved. For a rigid manipulator, the tip trajectory is completely defined by the trajectory of the joint. Effective control of the joint is equivalent to good control of the tip. The situation is not as straightforward for a flexible manipulator. The non-minimum phase property of the arm when the output is defined as the tip position makes exact asymptotic tracking of a desired tip trajectory impossible, if one is to employ causal controllers. Structural flexibility also causes extreme difficulty in modeling the manipulator dynamics. Furthermore, model truncation, which arises due to a finite-dimensional representation of a distributed parameter system causes unmodeled dynamics to be present in the mathematical model of the system. Further complications arise because of the highly nonlinear nature of the system and the difficulty involved in accurately modeling various friction and backlash terms. Moreover, coupling between the rigid dynamics and the flexible dynamics of the link may also cause stability problems. A change in the arm configuration and in the payload also leads to a change in the arm dynamics. This change has serious degrading effects on the performance of the manipulator controller.

Although linear controllers may yield desirable closed-loop performance, their region of operation is limited due to the nonlinearities present in the original system. The most common nonlinear approach to control a rigid manipulator is the inverse dynamics or computed torque strategy. This scheme assumes exact cancellation of the manipulator dynamics by the nonlinear controller. However, the extension of this approach to flexible-link manipulators is impeded by the non-minimum phase characteristic of the arm (the inverse dynamics controller will be unstable). To overcome this problem, decomposition of the inverse dynamics of the manipulator into a causal and an anti-causal systems was proposed in references [1, 2]. Alternatively, the output re-definition approach may be used to overcome the non-minimum phase characteristic of the system. In reference [3], the output of the system was re-defined as the "reflected" tip position (RTP) to ensure stable zero dynamics for the new input-output map. A re-defined output on the link between the joint and the tip was suggested.[4,5] The new output is defined so that the zero dynamics related to this output are stable. The singular perturbation approach, the integral manifold approach, and output-redefinition along with other conventional nonlinear control methods have also been used to control flexible-link manipulators.[6–12] In reference [13], a novel approach based on transmission zero assignment[14] was applied to control a single flexible-link manipulator.

However, all of these methods either assume exact knowledge of the dynamics (inverse dynamics), knowledge of the bounds on uncertainties (robust control), or knowledge of thc nonlinearities of the flexible-link system (adaptive control). Since, it is very difficult to model a flexible-link manipulator accurately, especially in the

* Dept. of Electrical Engineering, AmirKabir University, Tehran (Iran) 15914
** Dept. of Electrical and Computer Engineering, Concordia University, Montreal, Quebec (Canada) H3G 1M8
† Dept. of Electrical and Computer Engineering, University of Western Ontario, London, Ontario (Canada) N6A 5B9

https://doi.org/10.1017/S026357470200406X Published online by Cambridge University Press

presence of frictional-type nonlinearities, the performance of the above control strategies may be unsatisfactory for real applications.

The nonlinear mapping properties of neural networks, their adaptive nature, and their ability to deal with uncertainty make them a powerful tool for the control of flexible-link manipulators. Neural networks have been widely applied to the tracking control of *rigid* manipulators. Most of these results, however, rely on either minimum-phase characteristics of the input-output map of the system or on full state measurements, conditions that are generally not satisfied for flexible-link manipulators.

In reference [15], a neural network controller for a flexible-link manipulator was developed. Hub position and velocity measurements were used to stabilize the system and a neural observer/controller was proposed to drive the flexible arm to track a desired trajectory. This work, however, is restricted only to linear models of the flexible-link robot. In reference [6], partial knowledge of the dynamics of the flexible-link is assumed *a priori* and the unknown part of the dynamics is identified by a supervised learning algorithm. The control is constructed in two stages, an optimal controller and an unsupervised neural network controller using model-based predictive control. The scheme is based on an identification stage that also requires feedback from the states of the system. The model-based predictive control was also used[17] for controlling a flexible-link robot. In reference [18], the joint tracking control problem for a space manipulator is developed. However, tip position tracking cannot be guaranteed especially for highspeed desired trajectories.

In reference [19], the set-point regulation problem of a rigid manipulator holding a flexible plate was considered. The simplified model of the manipulator was used and based on the feedback-error-learning method, four neural networks were designed to serve only as feedforward controllers. In reference [20], tracking control of a partially known flexible-link robot was considered. The hub angle was selected for output tracking and control. The controller was composed of a singular perturbation-based fast control and an outer-loop slow control. The slow subsystem was controlled by a neural network for feedback linearization, plus a PD outer-loop for tracking and a robustifying term to ensure closed-loop stability. The methodology was later implemented on a single-link flexible manipulator.[21] Experimental results for very small movements of the arm were reported. This work is however, restricted to a manipulator with sufficiently large stiffness. In reference [22], the neural controller proposed in reference [23] was extended to a lightly damped system by adding to the cost function of the neural network, a term related to the hub velocity.

In reference [24], a linear model of the system was used for the purpose of only defining a new output for feedback control. The "feedback-error-learning" approach was then used to design a neural network controller that learns the nonlinear inverse dynamics of the flexible-link system. The results of reference [24] were extended to the case of multi-link manipulators in reference [25] where no *a priori* knowledge about the payload mass was assumed in defining the new output. The assumption of *a priori* knowledge about the linear model of the system was later relaxed by proposing two new schemes.[25–27] In the present paper, the two schemes proposed there are further improved to enhance the transient and steady-state response of the system, and to increase the dynamic range of the controller despite changes in both magnitude and frequency of the desired trajectories. This is accomplished by introducing a PD joint controller (without any *a priori* knowledge about the system dynamics) that helps to stabilize the closed-loop system in the early stages of the learning process. Experimental results verify and validate that this modification results in significant improvements in the system response.

2. DYNAMIC MODELING

Although an explicit information regarding the dynamic model of the manipulator is not utilized in designing our proposed neural network paradigms, the standard dynamic model of a flexible-link manipulator may be defined using the recursive Lagrangian approach[28]:

$$M(q)\ddot{q}+h(q,\dot{q})+Kq+F\dot{q}+F_c(q)=u \tag{1}$$

where $u=[\tau^T\ 0_{1\times m}]^T$, τ is the input torque, $q=[\theta^T\ \delta^T]^T$, θ *is the* $n\times 1$ vector of joint variables, δ *is the* $m\times 1$ vector of deflection variables, $h=[h_1(q,\dot{q})\ h_2(q,\dot{q})]^T$represents the terms due to gravity, Coriolis, and centrifugal forces, M is the mass matrix,

$$K=\begin{bmatrix} 0_{n\times n} & 0_{n\times m} \\ 0_{m\times n} & K_{m\times m} \end{bmatrix}$$

is the stiffness matrix, $F=diag\{F_1, F_2\}$ represents viscous friction at the hub and in the link structure, and $F_c=[f_c^T\ 0_{1\times m}]^T$ is the hub friction. Although, hub friction cannot be modeled very accurately, f_c is often considered as Coulomb friction which may be represented by a hard nonlinearity $f_c=C_{coul}SGN(\dot{\theta})$, where

$$SGN(\dot{\theta})=\begin{cases} 1 & \text{for } \dot{\theta}>0, \\ -1 & \text{for } \dot{\theta}<0, \\ 0 & \text{for } \dot{\theta}=0 \end{cases} \tag{2}$$

or may further be approximated, as in this paper, by a smooth sigmoidal function expressed as

$$f_c=C_{coul}\left(\frac{2}{1+e^{-a\dot{\theta}}}-1\right), \quad C_{coul}>0, \quad a>0.$$

As pointed out in reference [3], the tip position can be obtained from

$$y_{ti}=\theta_i+\frac{W_i(l_i,t)}{l_i},$$

https://doi.org/10.1017/S026357470200406X Published online by Cambridge University Press

where $W_i(l_i, t)$ is the elastic deflection at the tip and l_i is the length of the *ith* link. When m_i modes are considered for the *ith* link, $W_i(l_i, t)$ may be expressed as

$$W_i(l_i, t) = \sum_{j=1}^{m_i} \phi_{ij}(l_i)\delta_{ij}(t),$$

where ϕ_{ij} is the *jth* eigenfunction of the *ith* link and δ_{ij} is the *jth* mode of the *ith* link, respectively.

3. PROPOSED CONTROL STRUCTURES

This section presents two neural network-based control structures for tip-position tracking control of a flexible-link manipulator. The control structures developed in this section assume no *a priori* knowledge about the dynamics of the flexible manipulator.

3.1. Background and motivation

Feedback-error-learning was first introduced in reference [29] and later was modified in reference [30]. Two adaptive learning control schemes were proposed using feedback-error-learning for neural network controllers. In both schemes, the error signal for the neural network is the output of the feedback controller. The advantage of this learning scheme is that the target signal or the desired output signal for the neural network is not required. Also, back propagation of the error (training) signal through the controlled system or through the model of the controlled system is not required. Since the feedback torque is chosen as the error signal for this learning strategy, the feedback torque is expected to decrease with learning. This implies that the error between the desired trajectory and the actual trajectory is required to tend to zero as learning evolves. The learning scheme is designated as "feedback-error-learning" to emphasize the importance of using the feedback torque as the error signal.

The idea of feedback-error-learning has been applied to control a rigid-link robot manipulator[29,30] where the system has no zero dynamics and all the state variables are available. For a flexible-link manipulator, however, the zero dynamics related to the tip are unstable and full state feedback is not available. As mentioned earlier, a certain output can be defined so that the zero dynamics corresponding to this new output are stable. This output can be defined as the joint variable, but it will not yield an acceptable tip response for a relatively flexible robot. It is shown that by using a new output (to be defined later in this section), the concept of feedback-error-learning may be invoked to control the tip position of a flexible-link manipulator.

Let us first summarize our previous results on which the main results of this paper are based. In references [25–27], two schemes were proposed for controlling a flexible-link manipulator. They are briefly reviewed below.

3.1.1. Original joint-based control (JBC) scheme. In this section, a control structure is developed by assuming no *a priori* knowledge about the linear model of the flexible manipulator. Since a linear model of the system is not available, a PD-type control law cannot be designed to stabilize the system, and a suitable output for feedback cannot be determined as before. In other words, the feedback-error-learning scheme cannot be applied directly.

The proposed control structure is shown in Figure 1. The key to designing this controller is to define the natural choice of joint position as the output for control which ensures the minimum phase property of the input-output map due to its colocated actuator/sensor pair and to damp out the tip elastic deformation by adding $W(l, t)$ in the cost function of the neural network. Consequently, the objective function for training the neural network is selected as $J = \frac{1}{2}(e^T K_1 e + \dot{e}^T K_2 \dot{e} + W(l, t)^T K_3 W(l, t))$, where $e := y_r - \theta$, y_r is the reference tip position trajectory, and $W(l, t)$ is introduced to reduce the vibrations of the flexible modes of the system. The objective function J is a weighted function of e, $\dot{e}$ and $W(l, t)$ where the corresponding weights are specified by K_1, K_2 and K_3. Note that higher order derivatives of the error can also be included in the objective function. The inputs to the network are θ, $\dot{\theta}$, and $W(l, t)$, and the output of the network is the control signal u. The weight adjustment mechanism is based on the steepest descent method, namely

$$\dot{w} = -\eta\left(\frac{\partial J}{\partial w}\right)^T,$$

where w is the vector of the weights of the network and η is the learning rate. Now $\dfrac{\partial J}{\partial w}$ is computed according to

$$\frac{\partial J}{\partial w} = \frac{\partial J}{\partial e}\frac{\partial e}{\partial w} + \frac{\partial J}{\partial \dot{e}}\frac{\partial \dot{e}}{\partial w} + \frac{\partial J}{\partial W}\frac{\partial W}{\partial w} = e^T K_1 \frac{\partial e}{\partial w} + \dot{e}^T K_2 \frac{\partial \dot{e}}{\partial w} + W^T K_3 \frac{\partial W}{\partial w}.$$

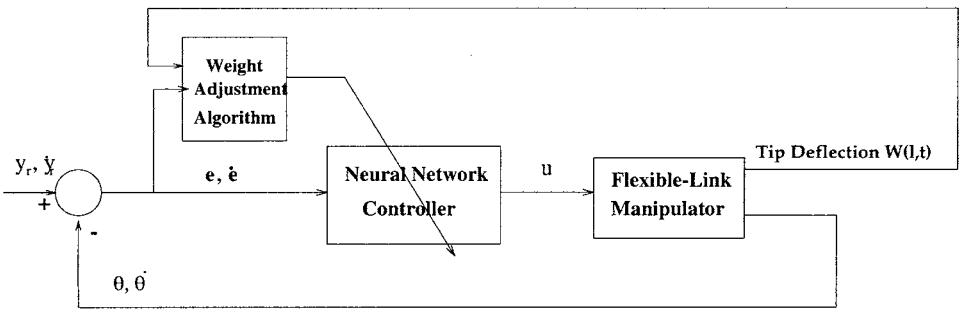


Fig. 1. Joint-based neural controller.

https://doi.org/10.1017/S026357470200406X Published online by Cambridge University Press

Since $e = y_r - \theta$, we get

$$\frac{\partial J}{\partial w} = -e^T K_1 \frac{\partial \theta}{\partial w} - \dot{e}^T K_2 \frac{\partial \dot{\theta}}{\partial w} + W^T K_3 \frac{\partial W}{\partial w},$$

and by using

$$\frac{\partial \theta}{\partial w} = \frac{\partial \theta}{\partial u}\frac{\partial u}{\partial w}, \frac{\partial \dot{\theta}}{\partial w} = \frac{\partial \dot{\theta}}{\partial u}\frac{\partial u}{\partial w}, \frac{\partial W}{\partial w} = \frac{\partial W}{\partial u}\frac{\partial u}{\partial w}$$

and $u = \Phi(e, \dot{e}, W, w)$, we may write $\frac{\partial J}{\partial w}$ as

$$\frac{\partial J}{\partial w} = \left(-e^T K_1 \frac{\partial \theta}{\partial u} - \dot{e}^T K_2 \frac{\partial \dot{\theta}}{\partial u} + W^T K_3 \frac{\partial W}{\partial u} \right) \frac{\partial \Phi}{\partial w},$$

where $\frac{\partial \Phi}{\partial w}$ can be computed using the backpropagation method, and

$$\frac{\partial \theta}{\partial u}, \frac{\partial \dot{\theta}}{\partial u}, \frac{\partial W}{\partial u}$$

are computed as suggested in reference [31] by using the sign of the gradient instead of its real value for training the neural controller.

However, controlling the joint position does not necessarily guarantee satisfactory response for the tip position due to the fact that there is no direct way to effectively damp out the elastic vibrations of the flexible modes at the tip. To remedy this limitation, the tip deflection was explicitly included in the objective function of the neural network resulting in improved transient response of the tip position.

3.1.2. Original output re-definition through online learning (ORTOL) scheme. It is well-known that the nonlinear flexible-link system is not in general input-state linearizable, however the system is locally input-output linearizable but the associated zero dynamics are unstable when the tip position is considered as the output.[32] To overcome this problem, the output of the system was re-defined in reference [3] as the "reflected" tip position (RTP) to ensure stable zero dynamics for the new input-output map. However, as the speed of the reference tip trajectory is increased, excitation of the flexible modes of the system makes the difference between the reflected tip position (RTP) and the actual tip position significant. Acceptable actual tip tracking performance cannot then be ensured by only controlling the RTP. Consequently, for very flexible-link manipulators, instead of the RTP, a different output should be defined for designing the feedback control law. A re-defined output on the link located between the joint and the tip was suggested in references [4, 5]. This new output may be defined as

$$y_{ai} = \theta_i + \alpha_i \frac{W_i(l_i, t)}{l_i}, \quad -1 \le \alpha_i \le 1, \tag{3}$$

where α_i is selected so that the zero dynamics associated to this output are stable. Note that $\alpha_i = -1$ corresponds to RTP, $\alpha_i = +1$ corresponds to the actual tip position, and $\alpha_i = 0$ corresponds to the hub angle. The objective here is to show that by using the new output y_a, the dynamics of the flexible-link manipulator may be expressed such that the new output re-definition approach is applicable for controlling the system.

Since, no *a priori* knowledge about the system dynamics was assumed, the appropriate output for feedback was re-defined using a neural network in an online learning scheme. This scheme, as shown in Figure 2, was denoted as output re-definition through online learning (ORTOL). In this scheme, two neural networks are employed. The first neural network (NN1) is trained to function as a feedback controller and the second neural network (NN2) is trained to provide a proper re-defined output to be used for feedback.

In other words, NN1 is trained to produce a control action so that the error between the output defined by NN2 and the desired reference trajectory is minimized. The objective function that is used for training NN1 is written as $J_1 = \frac{1}{2}(e^T K_1 e + \dot{e}^T K_2 \dot{e} + W(l, t)^T K_3 W(l, t))$, where $e := y_r - y_a$, y_a is constructed by measuring θ and adding to the output of NN2, and $W(l, t)$ is introduced to reduce the vibrations of

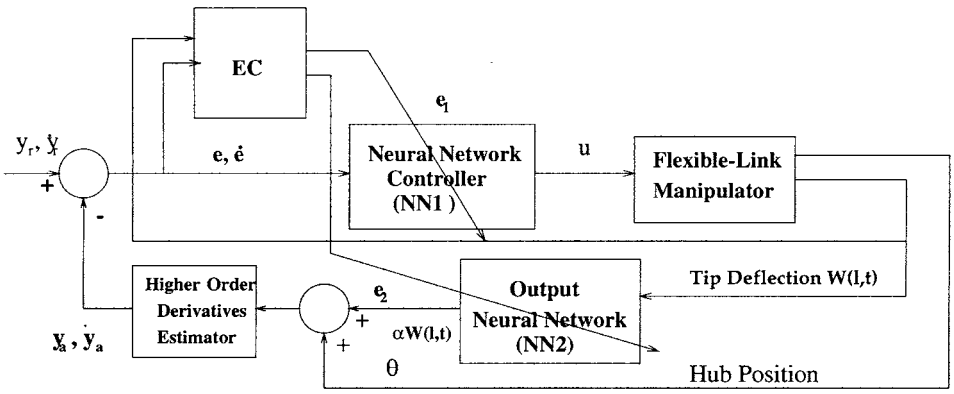


Fig. 2. Structure of the neural network based controller using output redefinition. The block EC performs the linear combination of e, $\dot{e}$, and $W(l, t)$ for specific learning schemes to be used in NN1 and NN2.

https://doi.org/10.1017/S026357470200406X Published online by Cambridge University Press

the flexible modes of the system in the output. The objective function J_1 is a weighted function of e, $\dot{e}$ and $W(l, t)$ where the corresponding weights are specified by K_1, K_2 and K_3. The inputs to the NN1 network are e, $\dot{e}$ and $W(l, t)$, and the output of the network is the control signal u. The weight adjustment mechanism is based on the steepest descent gradient method, namely

$$\dot{w} = -\eta_1 \left(\frac{\partial J_1}{\partial w} \right)^T,$$

$$\frac{\partial J_1}{\partial w} = \frac{\partial J_1}{\partial e}\frac{\partial e}{\partial w} + \frac{\partial J_1}{\partial \dot{e}}\frac{\partial \dot{e}}{\partial w} + \frac{\partial J_1}{\partial W}\frac{\partial W}{\partial w}$$

$$= e^T K_1 \frac{\partial e}{\partial w} + \dot{e}^T K_2 \frac{\partial \dot{e}}{\partial w} + W^T K_3 \frac{\partial W}{\partial w}.$$

Using $e = y_r - y_a$, we get

$$\frac{\partial J_1}{\partial w} = -e^T K_1 \frac{\partial y_a}{\partial w} - \dot{e}^T K_2 \frac{\partial \dot{y}_a}{\partial w} + W^T K_3 \frac{\partial W}{\partial w}.$$

Now, using

$$\frac{\partial y_a}{\partial w} = \frac{\partial y_a}{\partial u}\frac{\partial u}{\partial w}, \quad \frac{\partial \dot{y}_a}{\partial w} = \frac{\partial \dot{y}_a}{\partial u}\frac{\partial u}{\partial w}, \quad \frac{\partial W}{\partial w} = \frac{\partial W}{\partial u}\frac{\partial u}{\partial w}$$

and $u = \Phi(e, \dot{e}, W, w)$, we may write $\frac{\partial J_1}{\partial w}$ as

$$\frac{\partial J_1}{\partial w} = \left(-e^T K_1 \frac{\partial y_a}{\partial u} - \dot{e}^T K_2 \frac{\partial \dot{y}_a}{\partial u} + W^T K_3 \frac{\partial W}{\partial u} \right) \frac{\partial \Phi}{\partial w},$$

where $\frac{\partial \Phi}{\partial w}$ can be computed using the backpropagation method, and

$$\frac{\partial y_a}{\partial u}, \frac{\partial \dot{y}_a}{\partial u}, \frac{\partial W}{\partial u}$$

are computed as suggested in reference [31] by using the sign of the gradient instead of its real value for training the neural controller.

An approximation to the gradient was also suggested in reference [33] as

$$\frac{\partial y}{\partial u} = \frac{y(u + \delta u) - y(u)}{\delta u}. \tag{4}$$

This approximate derivative can be determined by changing each input to the plant slightly at the operating point and measuring the changes with previous iterations. Using the sign of gradient, the error of differentiation can be avoided.

The objective of the NN2 network is to generate an output of the form

$$y_{ai} = \theta_i + \alpha_i \frac{W_i(l_i, t)}{l_i}.$$

Since θ_i can be measured and $W_i(l_i, t)$ can be computed from y_{ti} and θ_i, a neural network can be trained to obtain an appropriate estimate for α_i. Using a network whose weights are limited to the range $[-1, 1]$, an objective function that can be minimized is selected as $J_2 = \frac{1}{2}(e^T e)$. This leads to

$$\dot{\alpha}_i = -\eta_2 \frac{\partial J_2}{\partial \alpha_i} = -\eta_2 e_i \frac{\partial e_i}{\partial \alpha_i} = \eta_2 e_i \frac{\partial y_{ai}}{\partial \alpha_i} = \eta_2 e_i W_i(l_i, t)$$

The input to the network is $W(l_i, t)$ and the output vector elements are computed as $\alpha_i W_i(l_i, t)$. The new output that is used for feedback is now constructed as

$$y_{ai} = \theta_i + \alpha_i \frac{W_i(l_i, t)}{l_i}.$$

3.2. The proposed new schemes

The control strategies developed in our previous work,[25–27] and briefly reviewed above assume no *a priori* knowledge about the system dynamics and no off-line training is required. Experimental results show that when no *a priori* knowledge about the system dynamics is assumed, in general smooth transient response is not always guaranteed. One intuitive way to overcome this problem is to use some off-line training to have a good "estimate" for the initial weights. However, this begs the question: How can one find an appropriate training set for the manipulator since the robot cannot move randomly in its workspace. Alternatively, we can use the simple approach that is described below.

To improve the transient as well as the steady-state response, we modify the previous two schemes introduced in references [25–27] by adding a joint PD controller to the neural network controllers. The joint PD control does facilitate stabilization of the closed-loop system in the early stages of the learning process, when the tracking error is large and changes in the weights are rapid. Consequently, it prevents the system from initially oscillating excessively and exciting other frequencies.

Consider the control law $u = u_c + u_n$ where u_c is a dissipative controller given by

$$u_c = -K_p \theta - K_v \dot{\theta}, \tag{5}$$

and u_n is the neural network-based controller designed as in references [25–27]. The addition of the controller u_c is motivated by the results developed in references [34, 35].

Theorem 3.1. *Assume K_p and K_v are symmetric and positive definite matrices. Then, the closed-loop system obtained by applying only (5) to the open-loop system (1) is locally asymptotically stable.*

The stability of the closed-loop system can be shown by selecting the following Lyapunov function and using LaSalle's theorem, namely

$$V = \frac{1}{2}\dot{q}^T M(q)\dot{q} + \frac{1}{2} q^T (K + B^T K_p B) q, \tag{6}$$

where $B = [I_{n \times n} 0_{n \times m}]$. For details refer to references [34, 35].

https://doi.org/10.1017/S026357470200406X Published online by Cambridge University Press

Since the result of this theorem is valid for any positive definite K_p and K_v, and does not require any *a priori* knowledge about the system dynamics, the controller u_c defined by (5) can be added to the output of the neural network controller u_n to guarantee a stable closed-loop system in the initial stages of the learning process and to improve the transient and steady state response of the system. Experimental results show that significant improvements in the system response may indeed be obtained by using this modification. In the following two subsections, we introduce our modification to the JBC and ORTOL schemes.

3.2.1. Modified JBC scheme. The modified JBC structure is shown in Figure 3. In this scheme, a joint PD controller and a neural network feedback controller are employed. The objective function for training the neural controller is considered as in the original JBC scheme as $J=\frac{1}{2}(e^TK_1e+\dot{e}^TK_2\dot{e}+W(l,t)^TK_3W(l,t))$. As with the original JBC scheme, $W(l,t)$ is included in the cost function J to damp out and minimize the elastic deflections at the tip. The gain K_3 specifies the weight placed on minimizing the elastic vibrations and the tracking error ability is influenced by the choice of gains K_1 and K_2. The inputs to the neural controller are θ, $\dot{\theta}$, and $W(l,t)$, and the output of the network is u_n. The weight adjustment rule used for the neural network is based on the steepest descent method, namely

$$\dot{w}=-\eta\left(\frac{\partial J}{\partial w}\right)^T,$$

where w is the weights vector of the network and η is the learning rate. As in Section 3.1.1 we can write $\frac{\partial J}{\partial w}$ as

$$\frac{\partial J}{\partial w}=\left(-e^TK_1\frac{\partial\theta}{\partial u}-\dot{e}^TK_2\frac{\partial\dot{\theta}}{\partial u}+W^TK_3\frac{\partial W}{\partial u}\right)\frac{\partial\Phi}{\partial w},$$

where $u_n=\Phi(\theta,\dot{\theta},W,w)$, with Φ denoting the nonlinearity that the neural controller is representing. The term $\frac{\partial\Phi}{\partial w}$ can be computed as in Section 3.1.1 by using the back-propagation method and

$$\frac{\partial\theta}{\partial u},\frac{\partial\dot{\theta}}{\partial u},\frac{\partial W}{\partial u}$$

are computed as suggested in reference [31] by using the sign of the gradient instead of its real value for training the neural controller. Note that the scalar product between the gradient produced by using the true value and that produced by using the sign is always positive, and will hence ensure error reduction.

3.2.2. Modified ORTOL scheme. The structure of the modified ORTOL scheme is shown in Figure 4. The joint PD control serves the same purpose as in the modified JBC case. In this structure, two neural networks are employed. The neural network controller NN1 is trained to produce a control law such that the tracking error between the desired reference trajectory y_r and the re-defined output y_a determined by the output of the neural network NN2 is minimized. The objective function tlrat is used for training NN1 is chosen as $J_1=\frac{1}{2}(e^TK_1e+\dot{e}^TK_2\dot{e}+W(l,t)^TK_3W(l,t))$, where $e:=y_r-y_a$. The inputs to the NN1 network are e, $\dot{e}$ and $W(l,t)$, and the output of the network is the control signal u_n. The weight adjustment rule is based on the steepest descent method, namely

$$\dot{w}=-\eta_1\left(\frac{\partial J_1}{\partial w}\right)^T,$$

where $\frac{\partial J_1}{\partial w}$ can be computed in a similar manner to that used for $\frac{\partial J}{\partial w}$ in Section 3.1.2.

The objective of NN2 is to adaptively learn to construct an appropriate re-defined output of the form

$$y_{ai}=\theta_i+\beta_i\frac{W_i(l_i,t)}{l_i},$$

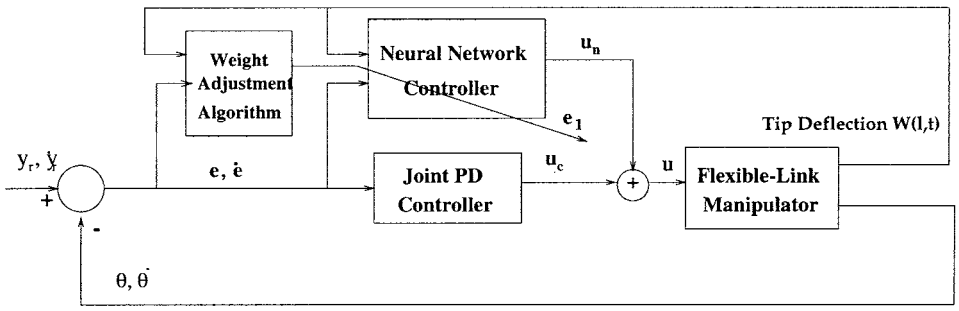


Fig. 3. The modified joint-based neural network controller.

https://doi.org/10.1017/S026357470200406X Published online by Cambridge University Press

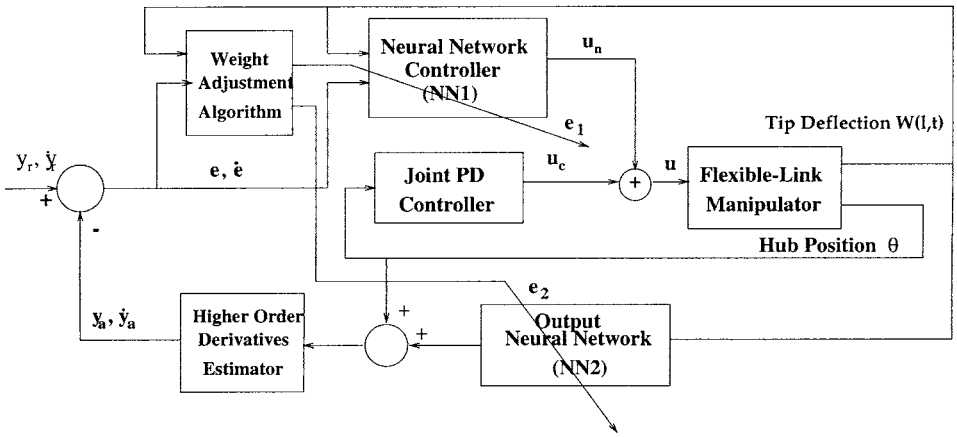


Fig. 4. Structure of the modified neural network based controller using output re-definition.

where β_i is the adjustable weight of the network. Since θ can be measured and $W(l, t)$ can be computed from y_t and θ, the neural network is basically trained to specify an appropriate estimate for α defined in (3). The network used here has its adjustable weight β_i limited to the range $[-1, 1]$ in order that the value of y_{ai} corresponds to the conventional definition used in the literature. The objective function to minimize is simply selected as $J_2=\frac{1}{2}(e^Te)$. Consequently, the adjustment law becomes

$$\dot{\beta}_i=-\eta_2\frac{\partial J_2}{\partial \beta_i}=-\eta_2 e\frac{\partial e}{\partial \beta_i}=\eta_2 e\frac{\partial y_{ai}}{\partial \beta_i}=\eta_2 e W_i(l_i, t)$$

The input to the network is $W_i(l_i, t)$ and the output vector elements are computed as $\beta_i W_i(l_i, t)$. The new re-defined output used for feedback is now simply constructed as

$$y_{ai}=\theta_i+\beta_i\frac{W_i(l_i, t)}{l_i}.$$

4. EXPERIMENTAL RESULTS

The experimental system consists of a highly flexible arm whose parameters are $l=1.2\ m$, $\gamma=1.2\ kg./m.$, $I_h=0.3\ kg.m.^2$, $EI=1.94\ Nt.m.^2$, $\omega_1=3\ rad/s$ and $\omega_2=19\ rad/s$, where l is the length of the link, γ is the mass per unit length, I_h is the hub inertia E is Young's modulus, I is the beam area moment of inertia, ω_j is the *jth* resonance frequency of the beam. For more details on the experimental test bed refer to reference [13]. A three-layer neural network was used for the neural controller in Figure 3 and NN1 of Figure 4 with 5 neurons in the hidden layer of each network. The hidden layer neurons have sigmoidal transfer functions and the output neuron uses a linear activation function. A single neuron whose weight is limited to the range $[-1, 1]$ was employed for the network NN2 as described in Section 3.2.2. The experimental results in this section are shown for only the actual manipulator tip position y_t and the results for y_a are not included. It is intuitively obvious that the tracking performance for the re-defined output y_a is better than that of y_t, however for practical reasons one is only interested in the behavior of y_t and not the "fictitious" output y_a.

In the first experirment, a $0.2+0.1\ sin(t)$ reference trajectory was applied to the manipulator with the nominal payload mass $M_l=30\ g$. Using the new output y_a defined as in (3) with α determined from [26, 27] as $\alpha=0.48$, a simple PD control law alone given by

$$u=k_p(y_r-y_a)+k_v(\dot{y}_r-\dot{y}_a),$$

with relatively large gains ($k_p=100$, $k_v=100$) was implemented and applied to the flexible arm. As expected, considerable amount of steady-state error in tracking the tip position was observed. The error is mainly caused by the presence of friction and stiction at the hub whose amplitudes vary with the hub position. The only way that PD control can overcome the effects of the friction and stiction at the hub is to have larger gains. As the PD gain k_p is increased, smaller steady state errors are obtained but at the expense of considerable oscillatory transient behavior of the tip. The result obtained by using $k_p=200$ and $k_v=100$ is shown in Figure 5-a. Note that the use of high gains also leads to instability of the closed-loop system for any step input that is greater than 0.2 *rad*. The response of the tip position to a 0.2 *rad* step input with the same PD gains is shown in Figure 5-b. In the next experiment, the payload mass M_l was increased to 850 *g* from the nominal value of 30 *g*. The response of the system to a 0.2 *rad* step input is shown in Figure 5-c. As can be observed, the response exhibits less oscillation but greater steady state error as connpared to the $M_l=30\ g$ case.

The original and the modified neural networks were implemented with the value of $K_1=1$, $K_2=1$, and $K_3=1$ in the objective functions J and J_1 and the learning rate of $\eta=0.02$. For the modified schemes, the PD joint controller was implemented as in (5) with the gains fixed at $K_p=30$, and $K_v=30$.

The responses of the actual tip position to a $0.2+0.1sin(t)$ desired trajectory for $M_l=30\ g$ are shown in Figure 5-d for the original JBC scheme and Figure 5-g for the original ORTOL scheme both introduced in references [26, 25, 27] and described in Section 3.1. The results clearly illustrate the improvements in the tracking performance achieved when compared to that obtained in Figure 5-a. Next, a 0.2 *rad* step input was applied to the system, and the

https://doi.org/10.1017/S026357470200406X Published online by Cambridge University Press

responses of the system are shown in Figure 5-e for the original JBC scheme and Figure 5-h for the original ORTOL scheme. These figures also clearly show improvements achieved in the responses of the controlled system when the neural network controllers are used. The robustness of the neural controllers was examined by increasing the payload mass M_l from 30 g to 850 g. The responses are shown in Figure 5-f for the original JBC scheme and Figure 5-i for the original ORTOL scheme. These figures further verify and demonstrate that the neural network controllers are more robust to the payload variations as compared to those obtained from conventional PD controller.

To demonstrate that further improvements in the response of the manipulator may be achieved, the modified schemes discussed in Section 3.2 were implemented and applied to the test-bed. For the sake of comparison, the previous experimental scenarios were repeated with the results shown in Figures 5-j to 5-o. The responses of the system to a $0.2+0.1sin(t)$ input are shown in Figure 5-j for the modified JBC scheme and Figure 5-m for the modified ORTOL scheme. It can be observed that the responses are significantly improved as compared to those obtained in Figures 5-d and 5-g. More specifically, much smoother transient and steady-state responses are obtained by using the modified schemes. Figure 6 shows the responses of the tip deflection to the $0.2+0.1sin(t)$ input for the original and the modified schemes. It can be observed that the vibration of the second fiexible mode of the system (19 *rad/s*) which exists in Figures 6-a (original JBC) and 6-b (original ORTOL) is damped out by using the modified schemes (refer to Figures 6-c (modified JBC) and 6-d (modified ORTOL)). To further verify the robustness of the proposed networks, the responses of the manipulator tip to a 0.2 *rad* step input are shown in Figure 5-k for the modified JBC scheme with M_l=30 g, Figure 5-n for the modified ORTOL scheme with M_l=30 g, Figure 5-l for the modified JBC scheme with M_l=850 g, and Figure 5-o for the modified ORTOL scheme with M_l=850 g. These results lead to the conclusion that, in general, using the proposed modified neural network controllers improves the system responses in terms of the following quantifiable metrics: (*i*) the overshoot of 2% for the modified JBC and 1% for the modified

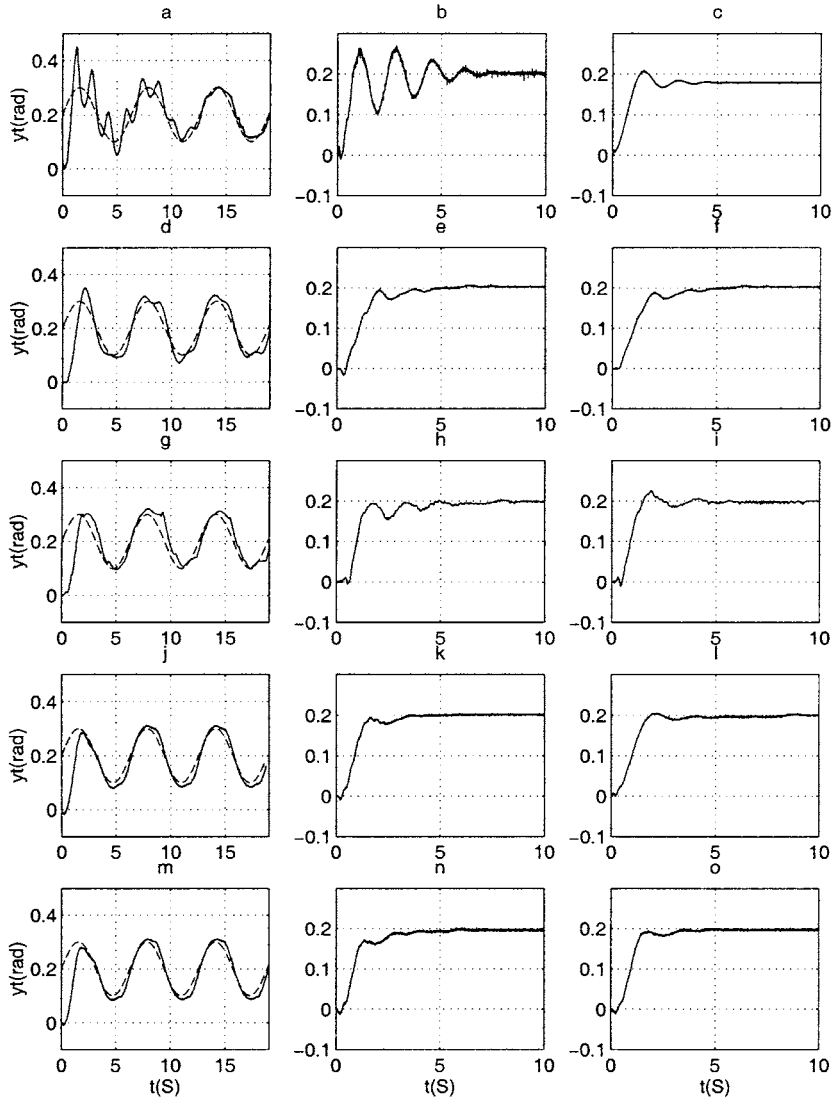


Fig. 5. Experimental system responses to sinusoidal and step inputs: (a,b,c) PD controller, (d,e,f) the original JBC neural controller, (g,h,i) the original ORTOL neural controller, (j,k,l) the modified JBC neural controller, and (m,n,o) the modified ORTOL neural controller (dashed lines correspond to the reference trajectories).

https://doi.org/10.1017/S026357470200406X Published online by Cambridge University Press

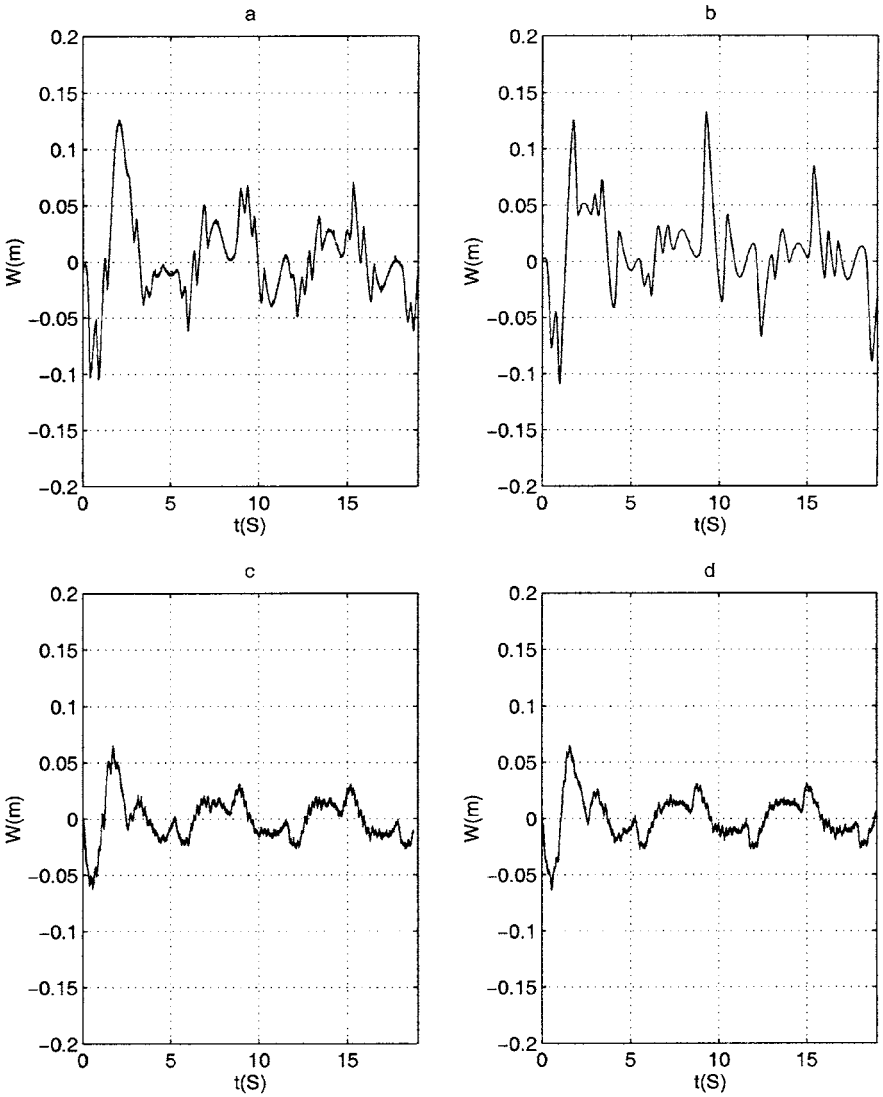


Fig. 6. The responses of the tip deflection to 0.2+0.1 *sin*(*t*): (a) the original JBC neural controller, (b) the original ORTOL neural controller, (c) the modified JBC neural controller, and (d) the modified ORTOL neural controller.

ORTOL as compared to 35% for the PD, 13% for the original ORTOL, and 4.5% for the original JBC, (*ii*) the undershoot of 5% for the modified JBC and 4% for the modified ORTOL as compared to 48% for the PD, 21% for the original ORTOL, and 13% for the original JBC, and (*iii*) the settling time of 3.3 *s* for the modified JBC and 3 *s* for the modified ORTOL as compared to 7.1 *s* for PD, 4.3 *s* for the original ORTOL, and 4.7 *s* for the original JBC. Note that in addition to the above results, the dynamic range and the robustness of the controller to payload variations are also improved.

One of the limitations of the original neural network controllers was their robustness to the changes in amplitude of the desired trajectories. Given that the original JBC and ORTOL schemes assume no *a priori* knowledge about the system dynamics, the flexible modes of the system at times vibrate with high amplitudes that generally depend on the input signal. In certain cases, the magnitude of the vibrations is so high that the tip deflections exceed the feasible range of the sensor measurements, leading to instability of the closed-loop system. For instance, the system becomes **unstable** when using the original JBC and ORTOL schemes for step inputs with amplitudes greater than 0.3 *rad*. However, using the modified JBC and ORTOL schemes, the system becomes more robust to the changes in the amplitude of the input signal and the dynamic range of the controller may be increased to as much as 0.6 *rad*. To demonstrate this, another set of experiments was performed with different input signals that have higher amplitudes. The responses of the manipulator to a 0.5 *sin*(*t*) input are shown in Figure 7-a for the modified JBC scheme and Figure 7-b for the modified ORTOL scheme. As expected, the modified ORTOL scheme exhibits less tracking error than the modified JBC scheme. The reason is that in the ORTOL scheme, the control is based on tracking a point closer to the tip, while in the JBC scheme the control is based on tracking the joint while the tip deflections are damped out simultaneously. Figures 7-c and 7-d show the responses of the system to a 0.5 *rad* step input for the modified JBC and ORTOL schemes, respectively. As can be seen from these results the

https://doi.org/10.1017/S026357470200406X Published online by Cambridge University Press

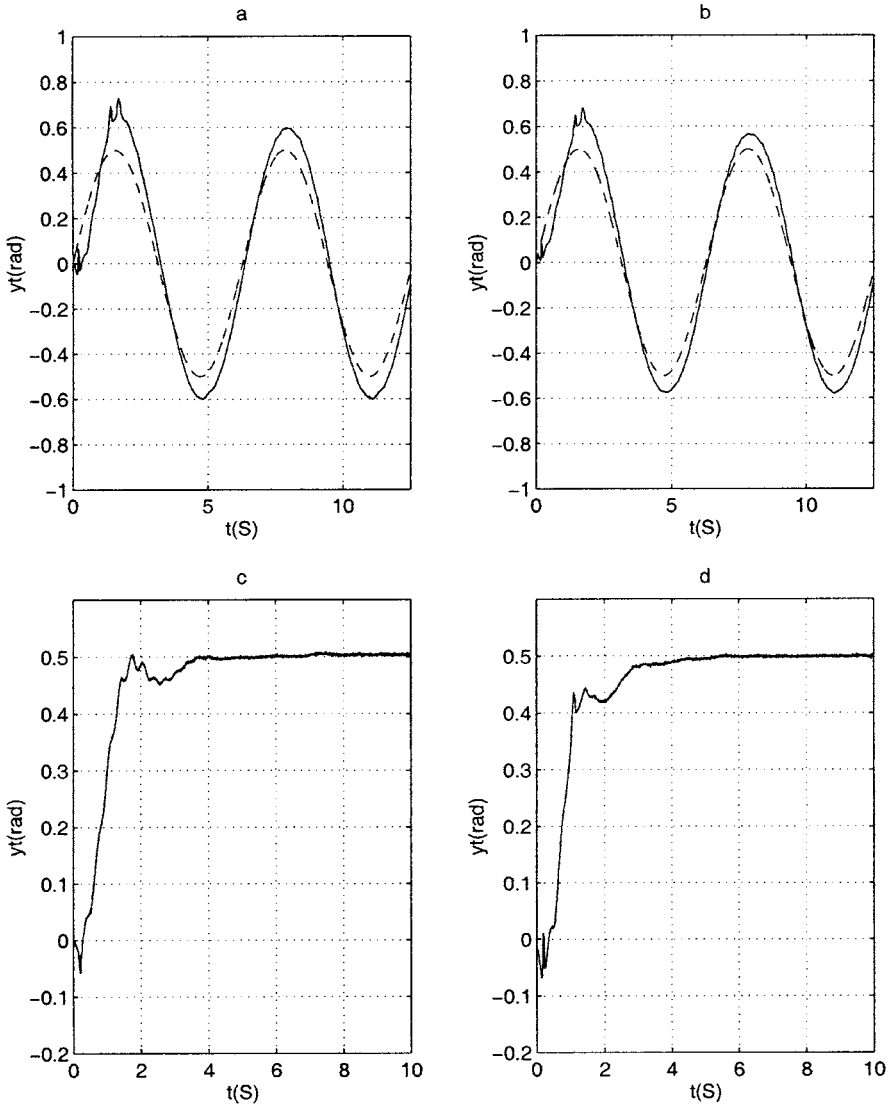


Fig. 7. Experimental system responses to 0.5 *sin*(*t*) and 0.5 *rad* step inputs with (a,c) the modified JBC neural controller, (b,d) the modified ORTOL neural controller (dashed lines correspond to the reference trajectories). The original JBC and ORTOL neural controllers lead to **instability** of the flexible manipulator.

modified schemes are more robust to the input signal variations.

5 CONCLUSIONS

In this paper, we have introduced two neural network-based schemes to control the tip-position of a flexible link manipulator. No *a priori* knowledge about the system dynamics is required in developiIlg the neural network controllers. The performance of the proposed neural network controllers was evaluated experimentally. The neural network controllers are all trained and employed as on-line controllers and no off-line training is needed for implementing them. Experimental results revealed the superiority of the proposed neural network controllers over a model-based PD controlleI and previously reported neural network controllers developed by the authors in references [25–27] in the presence of unmodeled dynamics, non-linearities such as hub friction and stiction, and payload variations.

This research was supported in part by Fonds pour la Formation de Chercheurs et l'Aide à la Recherche (FCAR) of the Province of Quebec under Grant ER-1042, and by NSERC (Canada) grants RGPIN-42515 and RGPIN-1345.

References

1. E. Bayo, "A finite-element approach to control the end-point motion of a single-link flexible robot", *J. Robotic Systems* 63–75 (1987).
2. O.S. Kwon and W.J. Book, "An inverse dynamic method yielding flexible manipulator state trajectories", *Proceedings of the American Control Conference* (1990) pp. 186–193.
3. D. Wang and M. Vidyasagar, "Transfer functions for a single flexible link", *Proceedings of the IEEE Int. Conf. on Robotics and Automation* (1989) pp. 1042–1047.
4. A. De Luca and B. Siciliano, "Trajectory control of a nonlinear one-link flexible arm", *Int. J. Control* **50**, No. 5, 1699–1715 (1989).

https://doi.org/10.1017/S026357470200406X Published online by Cambridge University Press

5. S.K. Madhavan and S.N. Singh, "Inverse trajectory control and zero dynamics sensitivity of an elastic manipulator", *Int. J. Robotics and Automation* 179–191 (1991).
6. M.A. Arteaga and B. Siciliano, "On Tracking Control of Flexible Robot Arms", *IEEE Trans. Automatic Control* **45**, No. 3, 520–527 (March, 2000).
7. K. Hashtrudi Zaad and K. Khorasani, "Control of non-minimum phase singularly perturbed systems with applications to flexible-link manipulators", *Proceedings of the Workshop on Advances in Control and its Applications* (H. Khalil, J. Chou and P. Ioannou, ed.), Lecture Notes in Control and Information Sciences 208 (Springer-Verlag, 1995) pp. 234–265.
8. M. Moallem, K. Khorasani and R.V. Patel, "An integral manifold approach for tip position tracking of flexible multi-link manipulators", *IEEE Transactions on Robotics and Antomation* **13**, No. 6, pp. 823–837 (December, 1997).
9. D.A. Schoenwald and U. Ozguner, "On combining slewing and vibration control in flexible manipulators via singular perturbations", *Proceedings of the 29th IEEE Conf. on Decision and Control* (1990) pp. 533–538.
10. B. Siciliano and W.J. Book, "A singular perturbation approach to control of lightweight flexible manipulators", *Int. J. Robotics Research* 79–90 (August, 1988).
11. B. Siciliano, W.J. Book and G. De Maria, "An integral manifold approach to control of a one-link flexible arm", *Proceedings of the 25th IEEE Conf. on Decision and Control* (1986) pp. 1131–1134.
12. H. Yang, H. Krishnan and M.H. Ang, "Tip-trajectory tracking control of single-link flexible robots via output redefinition", *Proceedings of the IEEE Int. Conference on Robotics and Antomation* (1999) pp. 1102–1107.
13. H. Geniele, R.V. Patel and K. Khorasani, "End-Point Control of a Flexible-Link Manipulator: Theory and Experiments", *IEEE Transactions on Control System Technology* **5**, No. 6, 556–570 (November, 1997).
14. R..V. Patel and P. Misra, "Transmission zero Assignment in Linear Multivariable Systems Part II: The General Case", *Proceedings of the American Control Conference* (1992) pp. 644–648.
15. W. Cheng and J.T. Wen, "A neural controller for the tracking control of flexible arms", *Proceedings of the IEEE Int. Conf. on Neural Networks* (1994) pp. 2327–2332.
16. J.D. Donne and U. Ozguner, "Neural control of a flexible-link manipulator", *Proceedings of the IEEE Int. Conf. on Neural Networks* (1994) pp. 2327–2332.
17. B. Song and A.J. Koivo, "Neural network model based control of a flexible link manipulator", *Proceedings of the IEEE Int. Conference on Robotics and Antomation* (1993) pp. 155–160.
18. R.T. Newton and Y. Xu, "Neural network control of a space manipulator", *IEEE Control Systems Magazine* 14–22 (December, 1993).
19. F. Arai, L. Rong and T. Fukuda, "Trajectory control of flexible plate using neural network", *Proceedings of the IEEE Int Conference on Robotics and Automation* (1993) pp. 155–160.
20. A. Yesildirek and F.L. Lewis, "Feedback linearization using neural networks", *Proceedings of IEEE Int. Conf. on Neural Networks* (1994) pp. 2539–2544.
21. L.B. Gutierrez, F.L. Lewis and J.A. Lowe, "Implementation of a Neural Network Tracking Controller for a Single Flexible Link: Comparison with PD and PID Controllers", *IEEE Trans. Industrial Electronics* **45**, No. 2, 307–318 (April, 1993).
22. A. Register, W.J. Book and C.O. Alford, "Artificial neural network control of a nonminimum phase, single-flexible-link", *Proceedings of IEEE Int. Conf. on Robotics and Automation* (1996) **Vol. 2**, pp. 1935–1940.
23. K. Takahashi and I. Yamada, "Neural-network-based learning control of flexible mechanism with application to a single link flexible arm", *Proceedings of the ASME WAM, Intelligent Control Systems* (1993) **Vol. 48**, pp. 95–104.
24. H.A. Talebi, K. Khorasani and R.V. Patel, "Experimental evaluation of neural network-based controllers for tip position tracking of a flexible-link manipulator", *Proceedings of the IEEE Inf Conf. on Robotics and Automation* (1997) pp. 3300–3305.
25. H.A. Talebi, K. Khorasani and R.V. Patel, "Tip-position tracking for flexible-link manipulators using artificial neural networks: Experimental results", *Proceedings of the IEEE Int. Joint Conference on Neural Networks* (1998) pp. 2063–2068.
26. H.A. Talebi, K. Khorasani and R.V. Patel, "Neural nctwork based control schemes for flexiblclink manipulators: Simulations and experiments", *Neural Networks, the 1998 Special Issue, Neural Control and Robotics: Biology and Technology* 1357–1377 (1998).
27. H.A. Talebi, R.V. Patel and K. Khorasani, "Inverse dynamics control of flexible link manipulators using neural networks", *Proceedings of the IEEE Int. Conference on Robotics and Automation* (1998) pp. 806–811.
28. W.J. Book, "Recursive Lagrangian Dynamics of Flexible Manipulator Arms", *Int. J. Robotics Research* **3**, No. 3, 87–100 (Fall, 1984).
29. H. Miyamoto, M. Kawato, T. Setoyama and R. Suzuki, "Feedback-error-learning neural network for trajectory control of a robotic manipulator", *Neural Networks* **1**, 251–265 (1988).
30. H. Gomi and M. Kawato, "Neural network control for a closed-loop system using feedback-error-learning", *Neural Networks* **6**, 933–946 (1993).
31. G. Lightbody, Q.H. Wu and G.W. Irwin, "Control application for feedforward networks", *Neural Networks for Control* (Miller, T.W. *et al.*, eds) (MIT Press, Cambridge, MA., 1990) pp. 51–71.
32. D. Wang and M. Vidyasagar, "Feedback linearizability of multi-link manipulators with one flexible link", *Proceedings of the 28th IEEE Conf. on Decision and Control* (1989) pp. 2072–2077.
33. D. Psaltis, A. Sideris and A.A. Yamamura, "A multilayered neural network controller", *IEEE Control Systems Magazine* 17–21 (April, 1988).
34. A. De Luca and B. Siciliano, "An asymptotically stable joint PD controller for robot arms with flexible links under gravity", *Proceedings of the 31st IEEE Conf. on Decision and Control* (1992) pp. 325–326.
35. A.G. Kelkar, S.M. Joshi and T.E. Alberts, "Globally stabilizing controllers for flexible multibody systems", *Proceedings of the 31st IEEE Conf. on Decision and Control* (1992) pp. 2856–2859.

https://doi.org/10.1017/S026357470200406X Published online by Cambridge University Press