Model-Independent Control of a Flexible-Joint Robot Manipulator

Withit Chatlatanagulchai
Department of Mechanical Engineering,
Kasetsart University,
50 Phahon Yothin Road,
Chatuchak, Bangkok 10900, Thailand
e-mail: fengwtc@ku.ac.th

Peter H. Meckl
Mem. ASME
School of Mechanical Engineering,
Purdue University,
West Lafayette, IN 47907
e-mail: meckl@purdue.edu

Flexibility at the joint of a manipulator is an intrinsic property. Even "rigid-joint" robots, in fact, possess a certain amount of flexibility. Previous experiments confirmed that joint flexibility should be explicitly included in the model when designing a high-performance controller for a manipulator because the flexibility, if not dealt with, can excite system natural frequencies and cause severe damage. However, control design for a flexible-joint robot manipulator is still an open problem. Besides being described by a complicated system model for which the passivity property does not hold, the manipulator is also underactuated, that is, the control input does not drive the link directly, but through the flexible dynamics. Our work offers another possible solution to this open problem. We use three-layer neural networks to represent the system model. Their weights are adapted in real time and from scratch, which means we do not need the mathematical model of the robot in our control algorithm. All uncertainties are handled by variable-structure control. Backstepping structure allows input efforts to be applied to each subsystem where they are needed. Control laws to adjust all adjustable parameters are devised using Lyapunov's second method to ensure that error trajectories are globally uniformly ultimately bounded. We present two state-feedback schemes: first, when neural networks are used to represent the unknown plant, and second, when neural networks are used to represent the unknown parts of the control laws. In the former case, we also design an observer to enable us to design a control law using only output signals—the link positions. We use simulations to compare our algorithms with some other well-known techniques. We use experiments to demonstrate the practicality of our algorithms.
[DOI: 10.1115/1.3117185]

Keywords: flexible-joint robot, backstepping, intelligent control, nonlinear systems, output feedback, variable-structure control, neural networks

1 Introduction

Motion control of the flexible-joint robot manipulator is an interesting and practical problem for several reasons. First, joint flexibility exists in most manipulators. It arises from driving components such as actuators, gear teeth, or transmission belts. In some applications, the designers incorporate flexible joints into their products intentionally to absorb impact force and to reduce damage to the parts from accidental collision. Second, control designers should explicitly include joint flexibility in their design because joint resonant frequencies, which are located within the control bandwidth, can be excited and cause severe oscillations. The experiment conducted by Sweet and Good [1] suggested that designers should consider joint flexibility in both modeling and control design.

Controller design of a two-link flexible-joint robot manipulator is challenging for two main reasons. First, its Euler–Lagrange model is much more complicated than the model of a rigid-joint or one-link flexible-joint robot manipulator. Second, the number of degrees of freedom is twice the number of control inputs. The control inputs do not directly act on the links. Instead, the control inputs directly act on the motors that connect to the links via flexible-joint dynamics. This results in the loss of some important structural properties that apply for rigid-joint robot manipulators, such as the matching property between nonlinearities and the inputs, and passivity from inputs to link velocities.

Some well-established control designs have been developed for flexible-joint robot manipulators. Spong [2] transformed the dynamic model of the flexible-joint manipulator into the standard singular perturbation model, by using link position as the slow variable and joint torque as the fast variable. The controller is a composite of slow and fast control. The slow-control input, which adds damping to the system, drives the closed-loop system to a quasisteady-state system that has the structure of a rigid-joint manipulator. Then, the fast-control input can be designed using available techniques for the rigid-joint manipulator. To avoid having to measure the joint torque signal, Nicosia et al. [3] described how to design an observer. Ge [4] derived an alternative singular perturbation model by using the tracking error of the motor shaft as the fast variable. Ge et al. [5] extended the work of Ge [4] to the case where model uncertainties exist in the system. They used radial basis function networks to estimate unknown functions, and used a discontinuous variable-structure controller to provide the closed-loop system with robustness for the estimation errors.

Under the assumption that the kinetic energy of the motor is due mainly to its own rotation, the flexible-joint robot manipulator model is feedback linearizable by static feedback control laws, as in the work of Spong and Vidyasagar [6]. De Luca and Lucibello [7] relaxed this assumption, and applied the so-called dynamic feedback linearization method to a more general robot manipulator model.

Brogliato et al. [8] compared three types of controllers: a controller developed from a decoupled model, a backstepping controller, and a passivity-based controller. For the first type, they decoupled the robot manipulator model by using the filtered error of link position and motor position error as variables. They also discussed a backstepping controller when model parameters are unknown but can be made to appear linearly with respect to

Contributed by the Dynamic Systems, Measurement, and Control Division of ASME for publication in the Journal of Dynamic Systems, Measurement, and Control. Manuscript received March 8, 2007; final manuscript received February 18, 2009; published online April 29, 2009. Assoc. Editor: Ranjan Mukherjee. Paper presented at the 2005 ASME International Mechanical Engineering Congress (IMECE2005), Orlando, FL, November 5–11, 2005.

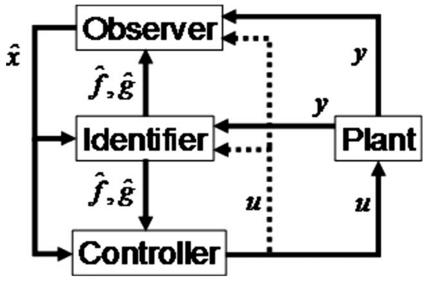

Fig. 1 Overall control system block diagram for the indirect method

known functions. The passivity-based controller was designed to shape the closed-loop total energy to a desired value to achieve passivity.

Some of the more recent works have been performed by Huang and Chen [9], Park [10], Subudhi and Morris [11], and Schaffer and Hirzinger [12]. Huang and Chen [9] presented experimental results on a one-link flexible-joint robot manipulator in the vertical plane. Park [10] used the feedback linearization method and a Takagi–Sugeno fuzzy system to replace model uncertainties. The work by Subudhi and Morris [11] contains good references on flexible-joint and flexible-link robot manipulators. The work by Schaffer and Hirzinger [12] contains some useful insights, which have been obtained from actually working with industrial robots.

Nevertheless, the control of a flexible-joint robot manipulator is still an open problem. Over the past three decades, many researchers have come up with various techniques to control the trajectory of a manipulator with flexible joints, but no single technique has proved practical and useful enough to be applied in all situations.

Under some reasonable assumptions, a flexible-joint robot manipulator's dynamic equations can be put in the nonlinear form suitable to our proposed control system. In that case, a controller can be configured so that its dynamic components are automatically tuned to the dynamics of the actual robot. We accomplish this by using neural networks to "learn" these dynamics in real time during operation.

In the work described in this paper, we have included both theoretical and experimental studies of a trajectory-tracking task of a two-link flexible-joint robot manipulator in the horizontal plane. The experiments have been performed on a robot manipulator in the Ruth and Joel Spira Laboratory, School of Mechanical Engineering, Purdue University. The work is divided into state-feedback control, when all states are available, and output-feedback control, when only link angular positions are available.

Two distinctly different control strategies are explored here. In the indirect method, three-layer neural networks are used to represent unknown plant functions, then, the control laws are designed based on the estimated plant functions. In the direct method, the control laws are designed first, then, three-layer neural networks are used to represent the unknown parts in the control laws. Figure 1 shows the overall block diagram of the control system for the indirect method. The following briefly describes how the overall system works. The observer is designed from identified plant functions from the identifier, control input from the controller, and actual output from the actual plant. The identifier then takes the estimated states from the observer and computes the identified plant functions. The controller uses estimated states from the observer and identified plant functions from the identifier in the control algorithm.

Our objective in designing a controller is to enable the output of the plant to track a desired trajectory. The closed-form mathematical model representing the plant is not required in the algorithm. However, to be able to design the controller, we need to assume that the actual plant is in a nonlinear state-space form. Three-layer neural networks are used to estimate the unknown plant functions.

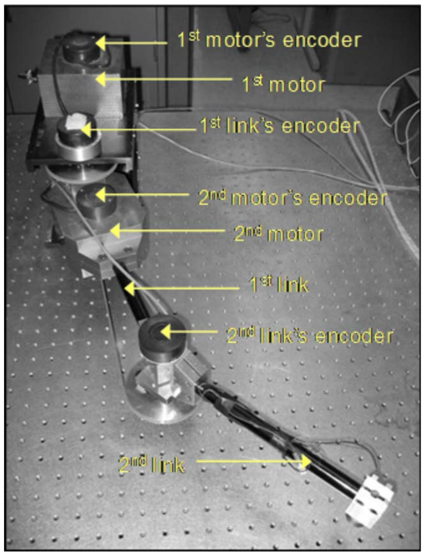

Fig. 2 Photograph of the two-link flexible-joint robot manipulator in the laboratory

When states are not measurable, an observer is added to the control system to estimate the unavailable states. Since uncertainties arise from estimation processes and external disturbances, variable-structure control is used as the robust controller to handle the uncertainties. The learning process of the neural network is performed online.

Using Lyapunov's second method, the design parameters including the neural networks' weight-update laws are designed so that the derivative of the Lyapunov function takes on some desired values. Lyapunov's second method is useful since it enables one to determine stability without explicitly finding the solution. Moreover, because uncertainties exist in the system, equilibrium points are difficult to find or even if they are found, they may not be located since they can be functions of uncertainties. Instead of using stability theorems for equilibrium points, we use boundedness theorems where boundedness of the error trajectory can be evaluated. Various analysis techniques can be found in the book by Khalil [13].

The paper is organized as follows. Section 2 describes the robot model and how it can be transformed into the applicable form. Section 3 contains state-feedback control design. Section 4 extends the work in Sec. 3 by adding an observer to estimate the state variables. Section 5 presents and compares simulation results of our algorithm with other techniques. Section 6 discusses the experimental setup and experimental results. The conclusions are given in Sec. 7.

2 Robot Model

Even if we do not use the robot model in our control algorithm, we still need to derive the robot model for two reasons. First, we need to show that this type of robot possesses a model that, under some mild assumptions, can be transformed into the strict-feedback form and therefore is applicable to our control system. Second, with the inclusion of actuator nonlinearities, such as deadzone and backlash, the model is rather complete and is used to represent the actual robot in the simulations in Sec. 5.

Figure 2 shows the two-link flexible-joint robot manipulator for which we are designing the controller. The manipulator operates in the horizontal plane and functions as follows. Input torque T_1 is applied to the first motor, which drives the first sprocket through a chain. The sprocket is attached to the first link via the first torsional spring, which provides joint flexibility. The second motor is situated on the first link. Input torque T_2 is applied to the second motor, which drives the second sprocket. The second sprocket is attached to the second link via the second torsional spring.

For the first link, θ_1 is the absolute angular position and, for the second link, θ_2 is the angular position relative to θ_1. For the first motor, θ_3 is the absolute angular position and, for the second motor, θ_4 is the relative angular position. For the first sprocket, $\theta_5=\theta_3/r$ is the absolute angular position and, for the second sprocket, $\theta_6=\theta_4/r$ is the relative angular position, where r is the gear ratio, which is the same for both joints. The four equations of motion can be obtained from the Euler–Lagrange method and can be expressed in the following format:

$$M(q_1)\ddot{q}_1 + V(q_1,\dot{q}_1)\dot{q}_1 + F_1(\dot{q}_1) + K_1(q_1 - q_2) = 0$$

$$J\ddot{q}_2 + F_2(\dot{q}_2) + B\dot{q}_2 - K_2(q_1 - q_2) = T \tag{1}$$

where $q_1=[\theta_1,\theta_2]^T$, $q_2=[\theta_5,\theta_6]^T=[\theta_3/r,\theta_4/r]^T$, $M(q_1)$ is the link inertia matrix, $V(q_1,\dot{q}_1)\dot{q}_1$ represents coriolis and centrifugal terms, K_1 and K_2 are joint flexibility matrices, J represents the inertia of motors and sprockets, B contains internal damping of the torsional springs, $F_1(\dot{q}_1)$ and $F_2(\dot{q}_2)$ are viscous friction vectors, and $T=[T_1,T_2]^T$ is the vector of applied motor torques.

2.1 Transforming Into the Strict-Feedback Form. By neglecting the kinetic energy from the rotation of the first link and assuming that the internal damping of the torsional springs is small and motor angular velocities are large compared with link angular velocities, we can convert Eq. (1) into strict-feedback form. Defining the state vectors as $x_1=q_1$, $x_2=\dot{q}_1$, $x_3=q_2$, and $x_4=\dot{q}_2$, we obtain a model in the strict-feedback form as follows:

$$\begin{aligned}\dot{x}_1 &= x_2\\ \dot{x}_2 &= f_2 + g_2x_3\\ \dot{x}_3 &= x_4\\ \dot{x}_4 &= f_4 + g_4T\\ y &= x_1\end{aligned} \tag{2}$$

where y is the (measured) vector of link positions, and

$$f_2 = -M(x_1)^{-1}[V(x_1,x_2)x_2 + F_1(x_2) + K_1(x_1)]$$

$$f_4 = -J^{-1}[F_2(x_4) + Bx_4 - K_2(x_1 - x_3)]$$

$$g_2 = M(x_1)^{-1}K_1, \quad g_4 = J^{-1}$$

2.2 Incorporating Actuator Nonlinearites. We use a friction model based on the work of Canudas et al. [14]

$$\frac{d\xi}{dt} = \dot{\theta} - \frac{|\dot{\theta}|}{g(\dot{\theta})}\xi$$

$$\sigma_0 g(\dot{\theta}) = F_c + (F_s - F_c)e^{-(\dot{\theta}/\nu_s)^2}$$

$$F(\dot{\theta}) = \sigma_0\xi + \sigma_1\frac{d\xi}{dt} + \sigma_2\dot{\theta} \tag{3}$$

where σ_0, σ_1, σ_2, F_c, F_s, and ν_s are unknown parameters usually obtained from experiment. ξ is the average deflection of the bristles at the microscale. $\dot{\theta}$ is the relative angular velocity between the two surfaces. F is friction torque that comprises Coulomb friction, Stribeck effect, and viscous friction.

We use a deadzone model, as given in the work of Tao and Kokotovic [15]

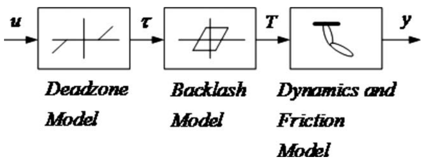

Fig. 3 Deadzone, backlash, and friction models are incorporated into the dynamic model of the two-link flexible-joint robot manipulator

$$\tau = D(u) = \begin{cases} m^-(u - d^-), & u \le d^- \\ 0, & d^- < u < d^+ \\ m^+(u - d^+), & u \ge d^+ \end{cases} \tag{4}$$

where d^-, d^+, m^-, and m^+ are unknown numbers.

We use a backlash model, as in Ref. [16]

$$\dot{T} = B(T,\tau,\dot{\tau}) = \begin{cases} m\dot{\tau} & \text{if } \dot{\tau} > 0 \text{ and } T = m(\tau - b^+) \text{ or} \\ & \text{if } \dot{\tau} < 0 \text{ and } T = m(\tau - b^-) \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

where b^-, b^+, and m are unknown numbers.

Figure 3 depicts the overall plant model where u is our designed control input, τ is the output of the deadzone model in Eq. (4), and T is the output of the backlash model in Eq. (5), which is the input torque that actually drives the manipulator. Both τ and T usually cannot be measured directly. Note that, even though deadzone and backlash change the magnitude of the designed control input u, the difference $\|u-T\|$ is bounded.

The friction terms in the robot model (2) are $F_1(\dot{q}_1) =[F(\dot{\theta}_1) \quad F(\dot{\theta}_2)]^T$ and $F_2(\dot{q}_2)=[F(\dot{\theta}_5) \quad F(\dot{\theta}_6)]^T$, where $F(\cdot)$ is the friction model given in Eq. (3).

3 State-Feedback Control Design

We design controllers assuming that the actual robot model is given by Eq. (2). To be certain that the controllers can handle external disturbances such as measurement noise or vibrations, we intentionally add additive disturbances to each subsystem of Eq. (2) to be

$$\begin{aligned}\dot{x}_1 &= x_2\\ \dot{x}_2 &= f_2(\bar{x}_2) + g_2(\bar{x}_2)(x_3 + d_{a2}(\bar{x}_4))\\ \dot{x}_3 &= x_4\\ \dot{x}_4 &= f_4(\bar{x}_4) + g_4(\bar{x}_4)(u + d_{a4}(\bar{x}_4))\\ y &= x_1\end{aligned} \tag{6}$$

where $\bar{x}_i=\{x_1,x_2,\ldots,x_i\}$ and $d_{ai}(\bar{x}_4)=[d_{ai1},d_{ai2}]^T$ are additive disturbances that may depend on all states.

The actuator nonlinearities including deadzone and backlash are added to model (6). Note from Fig. 3 that we actually design the control input u that goes into the deadzone model, not the actual torque T that drives the robot. Deadzone model (4) and backlash model (5) distinguish the designed control input u from the actual input T. And because they are unknown, they are treated as uncertainties.

3.1 Identifier. The diagram of a three-layer neural network is given in Fig. 4. Suppose a scalar-valued continuous function $g(z_1,z_2,\ldots,z_n):\mathbb{R}^n\to\mathbb{R}$ is to be approximated. The neural network has $z_1,z_2,\ldots,z_n,1$ as inputs. Variables in the network can be defined as follows:

$$\bar{Z} = [z_1,z_2,\ldots,z_n,1]^T \in \mathbb{R}^{n+1}$$

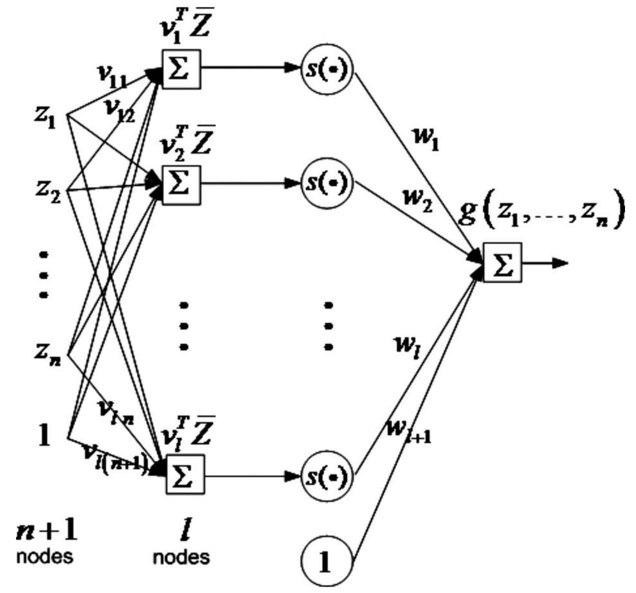

Fig. 4 A three-layer neural network

$$V=[v_1,v_2,\ldots,v_l]\in\mathbb{R}^{(n+1)\times l}$$

$$v_i=[v_{i1},v_{i2},\ldots,v_{i(n+1)}]^T\in\mathbb{R}^{n+1},\quad i=1,2,\ldots,l$$

$$S(V^T\bar{Z})=[s(v_1^T\bar{Z}),s(v_2^T\bar{Z}),\ldots,s(v_l^T\bar{Z}),1]^T\in\mathbb{R}^{l+1}$$

$$W=[w_1,w_2,\ldots,w_l,w_{l+1}]^T\in\mathbb{R}^{l+1}$$

$$g(W,V,z_1,z_2,\ldots,z_n)=W^TS(V^T\bar{Z})\in\mathbb{R}$$

$s(\bullet)$ can be any appropriate activation function that is a nonconstant, bounded, and monotonically increasing continuous function (see Theorem 3.1 in the book by Ge et al. [17]). In this work, a sigmoid function $s(z_i)=1/(1+e^{-z_i}),\forall z_i\in\mathbb{R}$ is used. This network is proved to be a universal approximator in a paper by Funahashi [18], which means that any continuous nonlinear function, $g(z_1,z_2,\ldots,z_n)$, can be approximated by a three-layer neural network with some constant ideal weight matrices, W^* and V^*, some appropriate number of hidden-layer nodes, l, with arbitrarily small approximation error. The function can be written as

$$g(z_1,z_2,\ldots,z_n)=W^{*T}S(V^{*T}\bar{Z})+\varepsilon$$

where $\|\varepsilon\|<\varepsilon_U$ is the approximation error with unknown $\varepsilon_U>0$, provided that $g(\cdot)$ is defined on a compact set Ω_z. Note that the foregoing statement is only to confirm the existence of ideal weights and ideal number of hidden-layer nodes. The appropriate number of hidden-layer nodes, in practice, can be found from trial and error. The ideal weights generally are unknown. However, in a system identification application, the ideal weights are typically assumed constant and bounded.

Usually when three-layer neural networks are used in control system design, the following two assumptions are required.

ASSUMPTION 1. *Any smooth nonlinear function* $g_i^*(\cdot)\in\mathbb{R}$ can be represented by a three-layer neural network with some constant ideal weights W_i^* and V_i^* as

$$g_i^*(\cdot)=W_i^{*T}S_i(V_i^{*T}\bar{Z}_i)+\varepsilon_i \tag{7}$$

where $\|\varepsilon_i\|<\varepsilon_{iU}$ is the approximation error with unknown $\varepsilon_{iU}>0$.

ASSUMPTION 2. *On the compact set* Ω_z, the ideal neural network weights W_i^* and V_i^* are constant and bounded by

$$\|W_i^*\|\le W_{iU},\quad \|V_i^*\|_F\le V_{iU}$$

where W_{iU} and V_{iU} are unknown.

Since ideal weights are unknown, let $\hat{W}$ and $\hat{V}$ be the estimates of W^* and V^*, respectively. The estimate of the function g is given by

$$\hat{g}(z_1,z_2,\ldots,z_n)=\hat{W}^TS(\hat{V}^T\bar{Z})$$

The three-layer neural network is considered a nonlinearly parametrized network since the weights v_{ij} appear nonlinearly. According to Barron [19], approximators that are nonlinear in their parameters can achieve the same level of approximation accuracy as those that are linear and usually require fewer number of adjusting parameters. However, since the parameters appear nonlinearly, the parameter-tuning law is usually more complicated.

In our control system design, the lack of knowledge of ideal weights is handled by the following lemma, which is taken directly from the book of Ge et al. [17]. The lemma approximates the difference between the neural network output with ideal weights and with estimated weights.

LEMMA 1. *Let* $\hat{W}$ and $\hat{V}$ be the estimates of W^* and V^*, respectively. Let the weight estimation errors be denoted by $\tilde{W}=\hat{W}-W^*$ and $\tilde{V}=\hat{V}-V^*$. Then, we have

$$\hat{W}^TS(\hat{V}^T\bar{Z})-W^{*T}S(V^{*T}\bar{Z})=\tilde{W}^T(\hat{S}-\hat{S}'\hat{V}^T\bar{Z})+\hat{W}^T\hat{S}'\tilde{V}^T\bar{Z}+d_u \tag{8}$$

where

$$\hat{S}=S(\hat{V}^T\bar{Z})\in\mathbb{R}^{l+1}$$

$$\hat{S}'=\mathrm{diag}\{\hat{s}_1',\hat{s}_2',\ldots,\hat{s}_l',0\}\in\mathbb{R}^{(l+1)\times(l+1)}$$

$$\hat{s}_i'=s'(\hat{v}_i^T\bar{Z})=\left.\frac{d[s(z_a)]}{dz_a}\right|_{z_a=\hat{v}_i^T\bar{z}}\in\mathbb{R},\quad i=1,2,\ldots,l$$

$$s(z_i)=1/(1+e^{-z_i}),\quad\forall z_i\in\mathbb{R}$$

The residual term d_u is bounded by

$$|d_u|\le\|V^*\|_F\|\bar{Z}\hat{W}^T\hat{S}'\|_F+\|W^*\|\|\hat{S}'\hat{V}^T\bar{Z}\|+|W^*|_1 \tag{9}$$

The symbol $\|\bullet\|_F$ denotes the Frobenius norm, that is, given a matrix A, the Frobenius norm is given by $\|A\|_F^2=tr(A^TA)=\Sigma_{i,j}a_{ij}^2$.

Proof. Use the Taylor series expansion of $S(V^{*T}\bar{Z})$ about $\hat{V}^T\bar{Z}$ and the facts that every element of the vector $\hat{S}-S(V^{*T}\bar{Z})$ is bounded by 1, $0\le s'(z_a)<0.25$, and $|z_as'(z_a)|\le 0.2239,\forall z_a\in\mathbb{R}$. The proof can be done as shown in Chap. 3 of the book by Ge et al. [17]. QED

Note that $\tilde{W}$ and $\tilde{V}$ appear linearly in Eq. (8). This is important since from Assumption 2, $\dot{\tilde{W}}=\dot{\hat{W}}$ and $\dot{\tilde{V}}=\dot{\hat{V}}$, therefore the weight adaptation laws can be designed using this linear structure.

3.2 Indirect Control Method. In this case, three-layer neural networks are used to represent the unknown plant functions, and control laws are designed based on these estimated plant functions. The following assumptions are required to design this type of controller.

ASSUMPTION 3. *The additive disturbances* d_{aik}, where $i=2,4;k=1,2$, are bounded by $\|d_{aik}\|<d_{aikU}$, where d_{aikU} are unknown constants.

ASSUMPTION 4. *There exist known constants* $g_{ijkU}>0$ such that $\|g_{ijk}(\cdot)\|\le g_{ijkU}$, $\forall i=2,4,\forall j=1,2,\forall k=1,2$.

ASSUMPTION 5. *The desired trajectory* x_{1d} is smooth, that is, continuously differentiable.

ASSUMPTION 6. *There exist unknown constants* $T_{iU}>0$ such that

$\|T_i - u_i\| \le T_{iU}$, $\forall i = 1, 2$.

The control objective is to make output $y = x_1$ follow desired trajectory x_{1d} as closely as possible, while all the signals in the closed-loop system remain bounded. For convenience, we drop arguments of some functions where appropriate.

In backstepping design, we try to reduce the error between actual state and desired state of each subsystem. The tracking error is the error of the first subsystem. Let $e_i = [e_{i1}, e_{i2}]^T = x_i - x_{id}$, $i = 1, \ldots, 4$ be those errors. We proceed with the following steps.

Step 1. Let the virtual control law of the first subsystem be

$$x_{2d} = -c_1 e_1 + \dot{x}_{1d} = [x_{21d}, x_{22d}]^T$$

where c_1 is a design parameter. The time derivative of the error of the first subsystem becomes $\dot{e}_1 = e_2 - c_1 e_1$. Note that Assumption 5 is required for the derivative of x_{1d} to exist.

Step 2. Let the virtual control of the second subsystem be

$$x_{3d} = -\hat{g}_2^{-1}[e_1 + c_2 e_2 + \hat{f}_2 - \dot{x}_{2d} - u_{3dvsc}] = [x_{31d}, x_{32d}]^T$$

From Eqs. (7) and (9) and Assumptions 3 and 4, we have

$$|d_{uf_{2j}}| + |\varepsilon_{f_{2j}}| + \sum_{k=1}^{2} \{|d_{ug_{2jk}} x_{3dk}|\} + \sum_{k=1}^{2} \{|\varepsilon_{g_{2jk}} x_{3dk}|\} + \sum_{k=1}^{2} \{|g_{2jk} d_{a2k}|\} \le K_{2j}^{*T} \phi_{2j}$$

where

$$K_{2j}^* = \Bigg[\|V_{f_{2j}}^*\|_F, \|W_{f_{2j}}^*\|, \|W_{f_{2j}}^*\|_1 + \varepsilon_{f_{2jU}} + \sum_{k=1}^{2} \{g_{2jkU} d_{a2kU}\}, \sum_{k=1}^{2} \{\|V_{g_{2jk}}^*\|_F\}, \sum_{k=1}^{2} \{\|W_{g_{2jk}}^*\|\}, \sum_{k=1}^{2} \{\|W_{g_{2jk}}^*\|_1\} + \sum_{k=1}^{2} \{\varepsilon_{g_{2jkU}}\} \Bigg]^T$$

$$\phi_{2j} = \Bigg[\|\bar{Z}_{f_{2j}} \hat{W}_{f_{2j}}^T \hat{S}'_{f_{2j}}\|_F, \|\hat{S}'_{f_{2j}} \hat{V}_{f_{2j}}^T \bar{Z}_{f_{2j}}\|, 1, \sum_{k=1}^{2} \{\|\bar{Z}_{g_{2jk}} \hat{W}_{g_{2jk}}^T \hat{S}'_{g_{2jk}} x_{3dk}\|_F\}, \sum_{k=1}^{2} \{\|\hat{S}'_{g_{2jk}} \hat{V}_{g_{2jk}}^T \bar{Z}_{g_{2jk}} x_{3dk}\|\}, \sum_{k=1}^{2} \{\|x_{3dk}\|\} \Bigg]$$

The variable-structure control law is given by $u_{3dvsc} = [u_{3dvsc1}, u_{3dvsc2}]^T \in \mathbb{R}^2$, where

$$u_{3dvscj} = -\hat{K}_{2j}^T \bar{\phi}_{2j}$$

$$\bar{\phi}_{2j} = \begin{bmatrix} \|\bar{Z}_{f2j} \hat{W}_{f2j}^T \hat{S}'_{f2j}\|_F \dfrac{2}{\pi} \arctan\left(\dfrac{e_{2j}}{\mu_{2j}} \|\bar{Z}_{f2j} \hat{W}_{f2j}^T \hat{S}'_{f2j}\|_F \right) \\ \hdashline \|\hat{S}'_{f2j} \hat{V}_{f2j}^T \bar{Z}_{f2j}\| \dfrac{2}{\pi} \arctan\left(\dfrac{e_{2j}}{\mu_{2j}} \|\hat{S}'_{f2j} \hat{V}_{f2j}^T \bar{Z}_{f2j}\| \right) \\ \hdashline \dfrac{2}{\pi} \arctan\left(\dfrac{e_{2j}}{\mu_{2j}} \right) \\ \hdashline \sum_{k=1}^{2} \{\|\bar{Z}_{g_{2jk}} \hat{W}_{g_{2jk}}^T \hat{S}'_{g_{2jk}} x_{3dk}\|_F\} \dfrac{2}{\pi} \cdot \\ \arctan\left(\dfrac{e_{2j}}{\mu_{2j}} \sum_{k=1}^{2} \{\|\bar{Z}_{g_{2jk}} \hat{W}_{g_{2jk}}^T \hat{S}'_{g_{2jk}} x_{3dk}\|_F\} \right) \\ \hdashline \sum_{k=1}^{2} \{\|\hat{S}'_{g_{2jk}} \hat{V}_{g_{2jk}}^T \bar{Z}_{g_{2jk}} x_{3dk}\|\} \dfrac{2}{\pi} \cdot \\ \arctan\left(\dfrac{e_{2j}}{\mu_{2j}} \sum_{k=1}^{2} \{\|\hat{S}'_{g_{2jk}} \hat{V}_{g_{2jk}}^T \bar{Z}_{g_{2jk}} x_{3dk}\|\} \right) \\ \hdashline \sum_{k=1}^{2} \{\|x_{3dk}\|\} \dfrac{2}{\pi} \arctan\left(\dfrac{e_{2j}}{\mu_{2j}} \sum_{k=1}^{2} \{\|x_{3dk}\|\} \right) \end{bmatrix} \tag{10}$$

μ_{2j} is a small positive design parameter, and $\hat{K}_{2j}$ approximates K_{2j}^*. The time derivative of the error of the second subsystem becomes

$$\begin{aligned} \dot{e}_2 &= \dot{x}_2 - \dot{x}_{2d} \\ &= \begin{bmatrix} \varepsilon_{f21} - \tilde{W}_{f21}^T (\hat{S}_{f21} - \hat{S}'_{f21} \hat{V}_{f21}^T \bar{Z}_{f21}) - \hat{W}_{f21}^T \hat{S}'_{f21} \tilde{V}_{f21}^T \bar{Z}_{f21} - d_{uf21} \\ \hdashline \varepsilon_{f22} - \tilde{W}_{f22}^T (\hat{S}_{f22} - \hat{S}'_{f22} \hat{V}_{f22}^T \bar{Z}_{f22}) - \hat{W}_{f22}^T \hat{S}'_{f22} \tilde{V}_{f22}^T \bar{Z}_{f22} - d_{uf22} \end{bmatrix} \\ &\quad + \left[\begin{array}{c:c} \varepsilon_{g211} - \tilde{W}_{g211}^T (\hat{S}_{g211} - \hat{S}'_{g211} \hat{V}_{g211}^T \bar{Z}_{g211}) - \hat{W}_{g211}^T \hat{S}'_{g211} \tilde{V}_{g211}^T \bar{Z}_{g211} - d_{ug211} & \varepsilon_{g212} - \tilde{W}_{g212}^T (\hat{S}_{g212} - \hat{S}'_{g212} \hat{V}_{g212}^T \bar{Z}_{g212}) - \hat{W}_{g212}^T \hat{S}'_{g212} \tilde{V}_{g212}^T \bar{Z}_{g212} - d_{ug212} \\ \hdashline \varepsilon_{g221} - \tilde{W}_{g221}^T (\hat{S}_{g221} - \hat{S}'_{g221} \hat{V}_{g221}^T \bar{Z}_{g221}) - \hat{W}_{g221}^T \hat{S}'_{g221} \tilde{V}_{g221}^T \bar{Z}_{g221} - d_{ug221} & \varepsilon_{g222} - \tilde{W}_{g222}^T (\hat{S}_{g222} - \hat{S}'_{g222} \hat{V}_{g222}^T \bar{Z}_{g222}) - \hat{W}_{g222}^T \hat{S}'_{g222} \tilde{V}_{g222}^T \bar{Z}_{g222} - d_{ug222} \end{array} \right] x_{3d} \\ &\quad - e_1 - c_2 e_2 + u_{3dvsc} + g_2 d_{a2} + g_2 (x_3 - x_{3d}). \end{aligned}$$

Step 3. Let the virtual control law of the third subsystem be

$$x_{4d} = -g_{2U} e_2 - c_3 e_3 + \dot{x}_{3d} = [x_{41d}, x_{42d}]^T$$

where

$$g_{2U} = \begin{bmatrix} g_{211U} & g_{212U} \\ g_{221U} & g_{222U} \end{bmatrix}$$

The time derivative of the error of the third subsystem is given by

$$\dot{e}_3 = -g_{2U} e_2 - c_3 e_3 + e_4$$

Step 4. Let the desired control law be

$$u = -\hat{g}_4^{-1}[e_3 + c_4 e_4 + \hat{f}_4 - \dot{x}_{4d} - u_{5dvsc}] = [u_1, u_2]^T$$

The time derivative of the error of the last subsystem, $\dot{e}_4$, can be derived similar to Step 2.

Since the designed control input u differs from the input torque T that actually drives the robot, there is an extra term, $g_4(T - u)$, in the $\dot{e}_4$ equation. However, the difference is bounded according to Assumption 6 and will be treated as an uncertainty, which will appear as an extra term $\Sigma_{k=1}^{2}\{g_{4jkU}T_{kU}\}$ in K_{4j}^{*}.

Since K_{4j}^{*} does not appear in the control law, the variable-structure control law, u_{5dvsc}, remains similar to Eq. (10) by replacing x_{3dk} with u_k, f_2 with f_4, and g_2 with g_4.

We use the following σ-modification weight-update laws:

$$\dot{\hat{W}}_{fij}=\Gamma_{wfij}[(\hat{S}_{fij}-\hat{S}'_{fij}\hat{V}_{fij}^{T}\bar{Z}_{fij})e_{ij}-\sigma_{wfij}\hat{W}_{fij}]$$

$$\dot{\hat{V}}_{fij}=\Gamma_{vfij}[\bar{Z}_{fij}\hat{W}_{fij}^{T}\hat{S}'_{fij}e_{ij}-\sigma_{vfij}\hat{V}_{fij}]$$

$$\dot{\hat{W}}_{gijk}=\Gamma_{wgijk}[(\hat{S}_{gijk}-\hat{S}'_{gijk}\hat{V}_{gijk}^{T}\bar{Z}_{gijk})x_{(i+1)dk}e_{ij}-\sigma_{wgijk}\hat{W}_{gijk}]$$

$$\dot{\hat{V}}_{gijk}=\Gamma_{vgijk}[\bar{Z}_{gijk}\hat{W}_{gijk}^{T}\hat{S}'_{gijk}x_{(i+1)dk}e_{ij}-\sigma_{vgijk}\hat{V}_{gijk}]$$

$$\dot{\hat{K}}_{lj}=\Gamma_{Klj}[\bar{\phi}_{lj}e_{lj}-\sigma_{Klj}\hat{K}_{lj}]$$

where $\Gamma_{wfij},\Gamma_{vfij},\Gamma_{wgijk},\Gamma_{vgijk},\Gamma_{Klj}>0$; $i=2,4$; $j=1,2$; $k=1,2$; and $l=1,\ldots,4$. The $\sigma>0$ terms in the update laws are design variables and are used to prevent $\hat{W}$, $\hat{V}$, and $\hat{K}$ from growing unboundedly by maintaining their values around their initial values.

Using the Lyapunov function

$$V=\sum_{i=2,4}\left\{\sum_{j=1}^{2}\sum_{k=1}^{2}\left[\tfrac{1}{2}\tilde{W}_{gijk}^{T}\Gamma_{wgijk}^{-1}\tilde{W}_{gijk}+\tfrac{1}{2}tr(\tilde{V}_{gijk}^{T}\Gamma_{vgijk}^{-1}\tilde{V}_{gijk})\right]+\sum_{j=1}^{2}\left[\tfrac{1}{2}\tilde{W}_{fij}^{T}\Gamma_{wfij}^{-1}\tilde{W}_{fij}+\tfrac{1}{2}tr(\tilde{V}_{fij}^{T}\Gamma_{vfij}^{-1}\tilde{V}_{fij})\right]\right\}+\sum_{i=1}^{4}\left[\tfrac{1}{2}e_i^{T}e_i+\sum_{j=1}^{2}\left(\tfrac{1}{2}\tilde{K}_{ij}^{T}\Gamma_{Kij}^{-1}\tilde{K}_{ij}\right)\right]$$

and the following facts:

$$2\tilde{W}^{T}\hat{W}=\|\tilde{W}\|^{2}+\|\hat{W}\|^{2}-\|W^{*}\|^{2}\geq\|\tilde{W}\|^{2}-\|W^{*}\|^{2}$$

$$2tr\{\tilde{V}^{T}\hat{V}\}=\|\tilde{V}\|_F^{2}+\|\hat{V}\|_F^{2}-\|V^{*}\|_F^{2}\geq\|\tilde{V}\|_F^{2}-\|V^{*}\|_F^{2}$$

$$2\tilde{K}^{T}\hat{K}=\|\tilde{K}\|^{2}+\|\hat{K}\|^{2}-\|K^{*}\|^{2}\geq\|\tilde{K}\|^{2}-\|K^{*}\|^{2}$$

$$0\leq|\alpha|-\alpha\frac{2}{\pi}\arctan\left(\frac{\alpha}{\mu}\right)\leq 0.2785\mu,\qquad\forall\,\alpha\in\mathbb{R}\tag{11}$$

and after some straightforward but lengthy derivation, we obtain the derivative of the Lyapunov function as

$$\dot{V}\leq-\varsigma V+\delta$$

where $\varsigma>0$ and $\delta\geq 0$. We refer the reader to Ref. [20] for more details and for the definitions of ς and δ.

From this point on, using standard nonlinear system analysis techniques, as described in the text by Khalil [13], it can be shown that the error trajectories, e, $\tilde{K}$, $\tilde{W}$, and $\tilde{V}$, are globally uniformly ultimately bounded.

3.3 Direct Control Method. In the direct method, the control laws are designed first, and three-layer neural networks are then used to represent the unknown parts in the control laws. Assumptions 3–6 are required to design this type of controller together with the following assumption.

ASSUMPTION 7. *The inverse matrices of* $g_i,\forall i=2,4$, *are positive definite.*

By letting $e_i=[e_{i1},e_{i2}]^{T}=x_i-x_{id}$, $i=1,\ldots,4$ be errors and proceeding with similar steps to those in the indirect method, we arrive at the following control laws:

$$x_{2d}=-c_1e_1+\dot{x}_{1d}=[x_{21d},x_{22d}]^{T}$$

$$x_{3d}=-e_1-c_2e_2-\begin{bmatrix}\hat{W}_{21}^{T}S_{21}(\hat{V}_{21}^{T}\bar{Z}_{21})\\ \hat{W}_{22}^{T}S_{22}(\hat{V}_{22}^{T}\bar{Z}_{22})\end{bmatrix}+u_{3dvsc}=[x_{31d},x_{32d}]^{T}$$

$$x_{4d}=-g_{2U}e_2-c_3e_3+\dot{x}_{3d}=[x_{41d},x_{42d}]^{T}$$

$$u=-e_3-c_4e_4-\begin{bmatrix}\hat{W}_{41}^{T}S_{41}(\hat{V}_{41}^{T}\bar{Z}_{41})\\ \hat{W}_{42}^{T}S_{42}(\hat{V}_{42}^{T}\bar{Z}_{42})\end{bmatrix}+u_{5dvsc}=[u_1,u_2]^{T}$$

and the weight-update laws

$$\dot{\hat{W}}_{ij}=\Gamma_{wij}[(\hat{S}_{ij}-\hat{S}'_{ij}\hat{V}_{ij}^{T}\bar{Z}_{ij})e_{ij}-\sigma_{wij}\hat{W}_{ij}]$$

$$\dot{\hat{V}}_{ij}=\Gamma_{vij}[\bar{Z}_{ij}\hat{W}_{ij}^{T}\hat{S}'_{ij}e_{ij}-\sigma_{vij}\hat{V}_{ij}]$$

$$\dot{\hat{K}}_{lj}=\Gamma_{Klj}[\bar{\phi}_{lj}e_{lj}-\sigma_{Klj}\hat{K}_{lj}]$$

where $\Gamma_{wij},\Gamma_{vij},\Gamma_{Klj}>0$; $i=2,4$; $j=1,2$; and $l=1,\ldots,4$.

4 Output-Feedback Control Design

Because the plant functions are unknown, we need to design an observer from the neural network estimated plant functions. Replacing the plant functions in Eq. (6) with the estimated functions and removing the additive disturbances, we have

$$\dot{\xi}_1=\xi_2$$

$$\dot{\xi}_2=\hat{f}_2(\bar{\xi}_2)+\hat{g}_2(\bar{\xi}_2)(\xi_3)$$

$$\dot{\xi}_3=\xi_4$$

$$\dot{\xi}_4=\hat{f}_4(\bar{\xi}_4)+\hat{g}_4(\bar{\xi}_4)(u)$$

$$\zeta=\xi_1$$

where ξ_i is the state vector of the estimated system, $\bar{\xi}_i=\{\xi_1,\xi_2,\ldots,\xi_i\}$, and ζ is the output of the system. $\hat{f}_i$ and $\hat{g}_i$ are vectors and matrices of estimated functions. $\hat{f}_{ij}$ and $\hat{g}_{ijk}$ are given as

$$\hat{f}_{ij}=\hat{W}_{fij}^{T}S_{fij}(\hat{V}_{fij}^{T}\bar{Z}_{fij})\in\mathbb{R},\quad\hat{g}_{ijk}=\hat{W}_{gijk}^{T}S_{gijk}(\hat{V}_{gijk}^{T}\bar{Z}_{gijk})\in\mathbb{R}$$

The mapping from actual states x_i to output derivatives is given by

$$y_e\overset{\Delta}{=}\begin{bmatrix}y_{e11}\\y_{e12}\\y_{e13}\\y_{e14}\\y_{e21}\\y_{e22}\\y_{e23}\\y_{e24}\end{bmatrix}=\begin{bmatrix}y_1\\\dot{y}_1\\\ddot{y}_1\\\dddot{y}_1\\y_2\\\dot{y}_2\\\ddot{y}_2\\\dddot{y}_2\end{bmatrix}\overset{\Delta}{=}H(\bar{x}_4)=\begin{bmatrix}x_{11}\\\varphi_{11}(\bar{x}_2)\\\varphi_{12}(\bar{x}_3)\\\varphi_{13}(\bar{x}_4)\\x_{12}\\\varphi_{21}(\bar{x}_2)\\\varphi_{22}(\bar{x}_3)\\\varphi_{23}(\bar{x}_4)\end{bmatrix}=\begin{bmatrix}H_1(\bar{x}_4)\\H_2(\bar{x}_4)\end{bmatrix}$$

where y_1 represents (measured) link 1 angle θ_1, and y_2 represents (measured) link 2 angle θ_2. Replacing the actual state x_i with the estimated state $\hat{x}_i$, we have mapping $\hat{H}(\bar{\hat{x}}_4)$ as

$$\hat{\zeta}_e \overset{\Delta}{=} \begin{bmatrix} \hat{\zeta}_{e11} \\ \hat{\zeta}_{e12} \\ \hat{\zeta}_{e13} \\ \hat{\zeta}_{e14} \\ \hat{\zeta}_{e21} \\ \hat{\zeta}_{e22} \\ \hat{\zeta}_{e23} \\ \hat{\zeta}_{e24} \end{bmatrix} \overset{\Delta}{=} \hat{H}(\bar{\hat{x}}_4) = \begin{bmatrix} \hat{x}_{11} \\ \psi_{11}(\bar{\hat{x}}_2) \\ \psi_{12}(\bar{\hat{x}}_3) \\ \psi_{13}(\bar{\hat{x}}_4) \\ \hat{x}_{12} \\ \psi_{21}(\bar{\hat{x}}_2) \\ \psi_{22}(\bar{\hat{x}}_3) \\ \psi_{23}(\bar{\hat{x}}_4) \end{bmatrix} = \begin{bmatrix} \hat{H}_1(\bar{\hat{x}}_4) \\ \hat{H}_2(\bar{\hat{x}}_4) \end{bmatrix}, \quad \hat{\zeta}_e \in \mathbb{R}^8$$

The nonlinear observer is given by

$$\begin{bmatrix} \dot{\hat{x}}_1 \\ \dot{\hat{x}}_2 \\ \dot{\hat{x}}_3 \\ \dot{\hat{x}}_4 \end{bmatrix} = \begin{bmatrix} \hat{x}_2 \\ \hat{f}_2(\bar{\hat{x}}_2) + \hat{g}_2(\bar{\hat{x}}_2)\hat{x}_3 \\ \hat{x}_4 \\ \hat{f}_4(\bar{\hat{x}}_4) + \hat{g}_4(\bar{\hat{x}}_4)u \end{bmatrix} + \left[\frac{\partial \hat{H}(\bar{\hat{x}}_4)}{\partial \bar{\hat{x}}_4}\right]^{-1} \varepsilon^{-1} L[y - \hat{\zeta}]$$

$$\hat{\zeta} = \hat{x}_1$$

where $\bar{\hat{x}}_i = \{\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_i\}$, $\varepsilon = \text{block-diag}[\varepsilon_1, \varepsilon_2] \in \mathbb{R}^{8\times 8}$, $\varepsilon_i = \text{diag}[\eta^1, \eta^2, \ldots, \eta^4]$, η is a design parameter with $0 < \eta \leq 1$, $[\partial \hat{H}(\bar{\hat{x}}_4)/\partial \bar{\hat{x}}_4]$ is the Jacobian of $\hat{H}$ with respect to $\hat{x}$, and $L = \text{block-diag}[L_1, L_2] \in \mathbb{R}^{8\times 2}$, where $L_i = [l_1, l_2, l_3, l_4]^T$ is such that $s^4 + l_1 s^3 + l_2 s^2 + l_3 s + l_4$ is a Hurwitz polynomial.

Using the observer above, it can be shown that the state estimation error, $\tilde{\zeta}_e = \hat{\zeta}_e - y_e$, is globally uniformly ultimately bounded. The proof is lengthy but can be found in Ref. [20].

The virtual controls and the actual control are given by

$$x_{2d} = -c_1 e_1 + \dot{x}_{1d} = [x_{2d1}, x_{2d2}]^T \in \mathbb{R}^2$$

$$x_{3d} = -\hat{g}_2^{-1}[g_{1U} e_1 + c_2 e_2 + \hat{f}_2 - \dot{x}_{2d} - u_{3dvsc}] = [x_{3d1}, x_{3d2}]^T \in \mathbb{R}^2$$

$$x_{4d} = -g_{2U} e_2 - c_3 e_3 + \dot{x}_{3d} = [x_{4d1}, x_{4d2}]^T \in \mathbb{R}^2$$

$$u = -\hat{g}_4^{-1}[g_{3U} e_3 + c_4 e_4 + \hat{f}_4 - \dot{x}_{4d} - u_{5dvsc}] = [u_1, u_2]^T \in \mathbb{R}^2$$

where $e_i = \hat{x}_i - x_{id}$, $\forall i = 1, \ldots, 4$. Note that we use estimated state $\hat{x}_i$ because the actual state is not available. The weight-update laws and the variable-structure control laws are the same as those of the state-feedback indirect control. The stability proofs also follow with minor modifications.

5 Simulation

We simulate two types of controllers that use the plant model in their control laws. They are computed torque control and model-based backstepping control. Computed torque control includes an inverse dynamics model of the robot as well as a proportional-integral-derivative (PID) feedback controller. The model-based backstepping control has the same algorithm as the indirect controller in Section 3; however, the unknown functions, f_i, g_i, $i = 2, 4$, are those obtained from system identification.

We also simulate four types of controllers that do not use the plant model in their control laws. They are PID control, indirect state-feedback backstepping intelligent control, direct state-feedback backstepping intelligent control, and output-feedback backstepping intelligent control. The last three control design techniques are the ones proposed in this paper.

Since the proposed techniques do not require the plant model in their control laws, the accuracy of the plant model does not affect their control performance. It is, therefore, interesting to see the performance of the model-independent control techniques when they are compared with the model-based control techniques, especially when uncertainties are present in the plant model. We present this in Sec. 5.1.

After the model-independent techniques are compared with the model-based techniques, the next interesting comparison would be among the model-independent techniques. We present this in Sec. 5.2.

Section 5.3 presents a comparison between the output-feedback backstepping intelligent control and the state-feedback backstepping intelligent control.

5.1 Model-Independent Versus Model-Based Methods. In this section, we compare the indirect state-feedback backstepping intelligent control and the direct state-feedback backstepping intelligent control with computed torque control and model-based backstepping control. Computed torque control and model-based backstepping control use the plant model in their control laws; therefore, their performance depends on the accuracy of the plant model.

Design parameters are as follows. For indirect state-feedback control, $l=3$, $\Gamma_{wf}=\Gamma_{vf}=\Gamma_{wg}=\Gamma_{vg}=10$, $\Gamma_k=1$, $c_i=5$, $\sigma_{wf}=\sigma_{vf}=\sigma_{wg}=\sigma_{vg}=\sigma_k=0.1$, and $\mu=1$. For direct state-feedback control, $\Gamma_{wij}=\Gamma_{vij}=\Gamma_{kij}=10$, $c_i=10$, $\sigma_{wij}=\sigma_{vij}=\sigma_{kij}=0.2$, and $\mu=0.1$. For computed torque control, which includes a PID controller, the control gains are $k_P=100$, $k_I=100$, and $k_D=100$. For model-based backstepping control, c_1, c_2, c_3, and c_4 are set to 3.0.

Because we want to focus on comparing controller performance, we remove the deadzone and backlash from the true model that represents the actual robot manipulator. We also remove external disturbances from the true model and set the force saturation limits to ± 1 Nm in all cases.

Figure 5 shows the tracking performance of the four controllers. In all cases, the plant model is intentionally corrupted to have its stiffness and damping at 80% of their actual values. When the plant model does not perfectly match the actual model, as is common in practice, we see the deterioration in the performance of the computed torque and model-based backstepping controllers. Note that the model uncertainties do not affect the performance of the direct and indirect backstepping controllers simply because they do not use the plant model in their algorithms. Moreover, the computed torque control performance seems to degrade more than that of the model-based backstepping control. This is because the computed torque control uses the plant model twice in its control algorithm. First, the plant model is used to compute desired motor trajectories from desired link trajectories. Second, the plant model is used as the inverse dynamics to cancel the nonlinearities in the actual model.

5.2 Comparison Among Model-Independent Techniques. Performance of PID control alone heavily depends on proper adjustment of the controller's gains. In this comparison, the following gains were used: $k_P=1$, $k_I=1$, and $k_D=1$. This leads to the tracking results shown in Fig. 6(*a*). When we change the amplitude of the desired trajectory from 0.6 rad to 1.25 rad and again apply the same PID control that produced the result in Fig. 6(*a*), we see that the system goes unstable, as shown in Fig. 6(*b*), which means the PID controller's gains must be redesigned. Another example is when we intentionally corrupt the actual robot model by multiplying the damping matrix by 0.8. Without redesigning, the PID control system went unstable, as shown in Fig. 6(*c*).

Meanwhile, the direct state-feedback control can handle the amplitude range from 0.6 rad to 1.25 rad and the change in damping without having to be redesigned, as shown in Fig. 6(*d*).

5.3 State-Feedback Versus Output-Feedback Controllers. To evaluate the tracking performance of our proposed controllers when there are uncertainties in the actual robot model, we let the external disturbances in Eq. (6) be

$$d_{a2} = [0.001 \text{ randn}(2,1)]^T$$

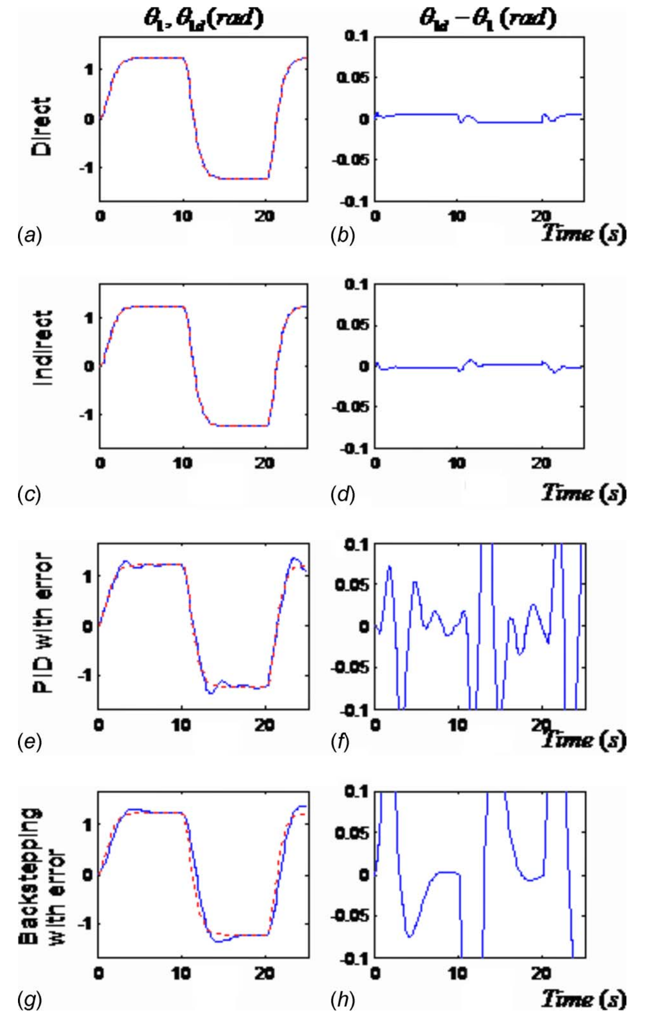

Fig. 5 Tracking performance comparison: (*a*) and (*b*) direct state-feedback, (*c*) and (*d*) indirect state-feedback, (*e*) and (*f*) computed torque, and (*g*) and (*h*) model-based backstepping

$$d_{a4} = [0.01\ \sin(\theta_1 \dot{\theta}_1), \arctan(\theta_1 \dot{\theta}_2)]^T$$

where "randn(2,1)" represents a 2×1 vector of pseudorandom numbers. We also included actuator nonlinearities, which are friction model (3), deadzone model (4), and backlash model (5), in the actual robot model.

For output-feedback control, we used the following parameters: $\Gamma_{wfi}=\Gamma_{vfi}=\Gamma_{wgi}=\Gamma_{vgi}=10$, $\Gamma_{ki}=1$, $c_i=15$, $\sigma_{wfi}=\sigma_{vfi}=\sigma_{wgi}=\sigma_{vgi}=\sigma_{ki}=0.1$, $\mu_{ij}=0.1$, $\eta=0.1$, $L_j=[16,91,216,180]^T$, $\forall i=2,4$, and $\forall j=1,2$.

Figure 7 contains the scatter plots showing deadzone and backlash. Figure 8 shows the first link's tracking performance in four cases: (a) using actual states and neural network estimated plant functions, (b) using estimated states and actual plant functions, (c) using actual states and actual plant functions, and (d) using estimated states and estimated plant functions. We see that the tracking performance of the proposed controllers (indirect state-feedback in case (a) and output-feedback in case (d)) is comparable to that of the ideal case (case (c)) when actual states and actual plant functions were used.

6 Experiment

6.1 Experimental Setup. Figure 2 depicts the two-link flexible-joint robot for which we have designed controllers. There are four optical encoders for two link and two motor positions.

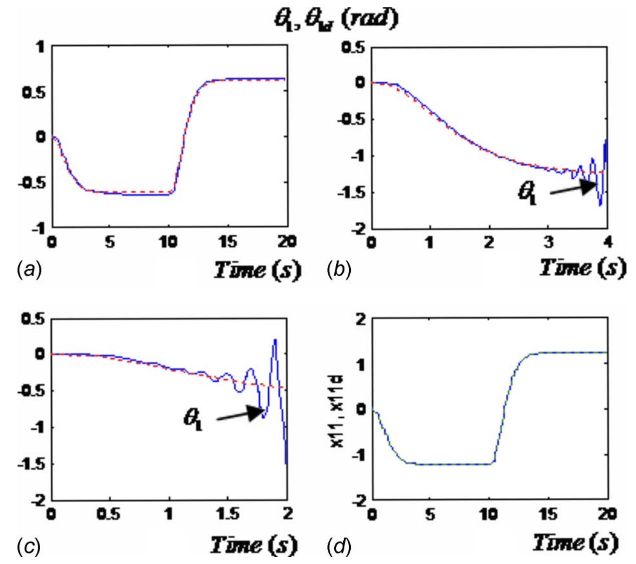

Fig. 6 Tracking performance: (*a*) PID with 0.6 rad amplitude, (*b*) PID with 1.25 rad amplitude, (*c*) PID with corrupted model, and (*d*) direct state-feedback with 1.25 rad amplitude and corrupted model

Angular velocities are obtained from the Euler method $\dot{\theta}_i(k+1) = [\theta_i(k+1) - \theta_i(k)]/t_s$, where t_s is sampling period. Two current amplifiers supply current to the two motors.

Figure 9 depicts the overall experimental setup. There are two desktop computers acting as host and target. The target computer contains the data acquisition board, receives the position signals from the encoders, and sends the controller output signals to the power amplifiers. The host computer runs the main software, which is used to interface with users and to monitor the activities in the target computer, activities such as reading the encoders, and the controller output signals. The host computer is also used to upload the program to the memory of the target computer.

Each encoder transmits two signals—the continuous trains of two square waves called channel A and channel B. The phase difference of the two signals is used to tell the direction of the rotation, whether clockwise or counterclockwise. The two signals connect to the data acquisition board through the digital input port. A software program is written to convert the two signals into angular position.

For the software, we use LABVIEW 7.1, LABVIEW REAL-TIME MOD-

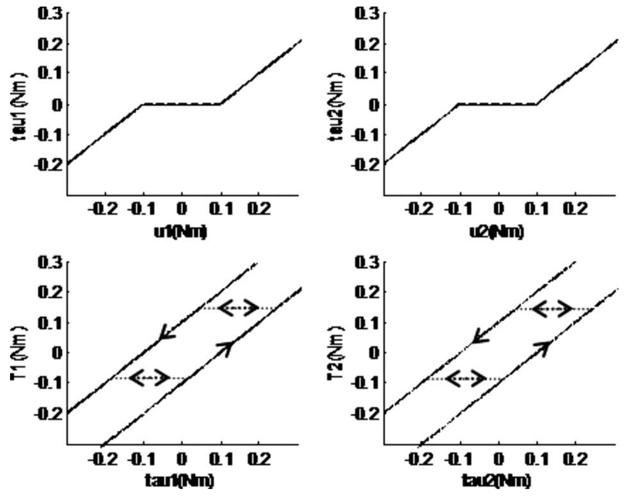

Fig. 7 Scatter plots showing deadzone and backlash

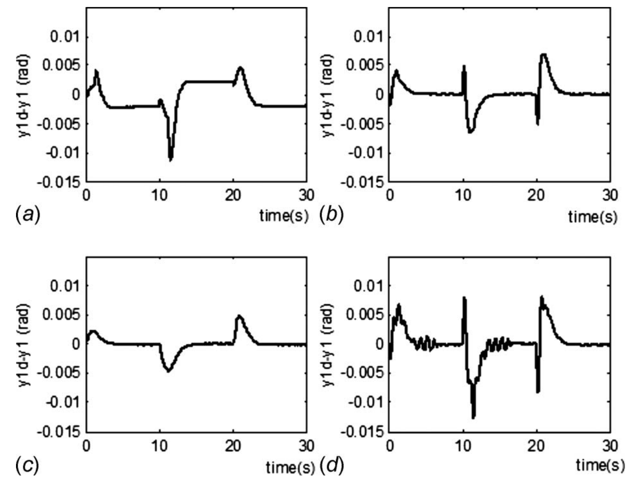

Fig. 8 Tracking error comparison: (*a*) actual states and estimated plant, (*b*) estimated states and actual plant, (*c*) actual states and actual plant, and (*d*) estimated states and estimated plant

ULE, and LABVIEW FPGA MODULE to perform hardware-in-the-loop experiments. The data acquisition board is National Instruments' PCI-7831R.

6.2 System Identification. For the purposes of generating a simulation model of the robot, and for the model-based control designs, system identification was performed using input and output signals from the actual robot. To focus only on getting the values of the robot parameters and to avoid the complexity from the discontinuous nonlinear terms, we did not incorporate the actuator nonlinearities in Eqs. (3)–(5) into the model when we performed the system identification. The parameter values of the friction model, the deadzone model, and the backlash model were obtained separately by approximation from the physical properties of the robot hardware.

The robot model (2) can be rearranged as a linear regression equation whose right-hand side contains known terms and whose left-hand side contains the product of known terms and unknown plant parameters. The unknown plant parameters were obtained

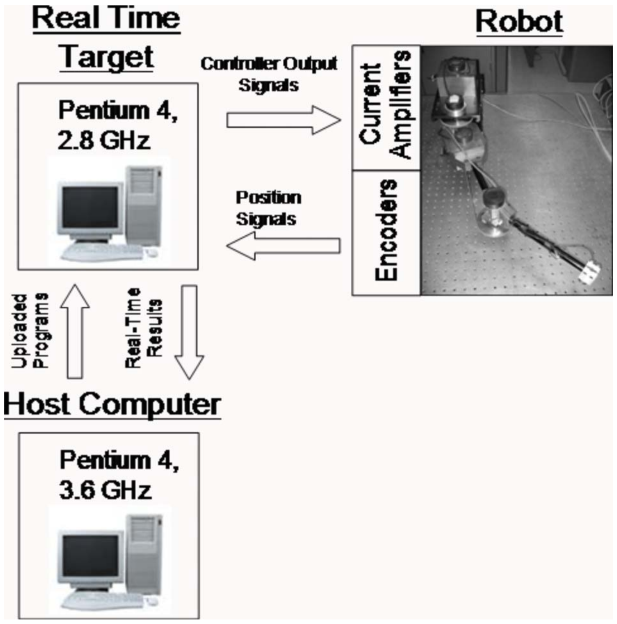

Fig. 9 Diagram showing overall experimental setup

from the linear least-square method. The reader can consult Ref. [20] for details of the system identification. The result is given by

$$M(q_1)=\begin{bmatrix} 0.201+0.06\cos\theta_2 & 0.0266+0.03\cos\theta_2 \\ 0.0266+0.03\cos\theta_2 & 0.0266 \end{bmatrix}$$

$$J=\begin{bmatrix} 0.017 & 0 \\ 0 & 0.014 \end{bmatrix}, \quad B=\begin{bmatrix} 5.66\times10^{-4} & 0 \\ 0 & 5.66\times10^{-4} \end{bmatrix}$$

$$V(q_1,\dot{q}_1)=\begin{bmatrix} 0 & -0.03(2\dot{\theta}_1+\dot{\theta}_2)\sin\theta_2 \\ 0.03\dot{\theta}_1\sin\theta_2 & 0 \end{bmatrix}$$

$$K_1=\begin{bmatrix} 0.4 & 0 \\ 0 & 0.4 \end{bmatrix}, \quad K_2=\begin{bmatrix} 0.075 & 0 \\ 0 & 0.075 \end{bmatrix}$$

Approximating from the physical properties of the robot hardware, the deadzone and backlash models have the following parameters. For deadzone model (4), we have $d^-=-0.1$, $d^+=0.1$, and $m^-=m^+=1$. For backlash model (5), we use $b^-=-0.1$, $b^+=0.1$, and $m=1$. The parameters in friction model (3) have the following values: $\sigma_0=0.01$, $\sigma_1=0.1$, $\sigma_{21}=0.02$ (for shoulder motor), $\sigma_{22}=0.056$ (for elbow motor), $F_c=10$, $F_s=20$, and $\nu_s=0.1$.

6.3 Experimental Results. We implemented direct and indirect state-feedback controllers using the following design parameters.

For the indirect state-feedback control,

$$\Gamma_{wij}=\Gamma_{vij}=\Gamma_{kij}=0.0001, \quad c_i=\begin{bmatrix} 13 & 0 \\ 0 & 29 \end{bmatrix}$$

$$\sigma_{wij}=\sigma_{vij}=\sigma_{kij}=0.1, \quad \mu=1$$

For the direct state-feedback control,

$$\Gamma_{wij}=\Gamma_{vij}=\Gamma_{kij}=0.0001, \quad c_i=\begin{bmatrix} 4.4 & 0 \\ 0 & 3.5 \end{bmatrix}$$

$$\sigma_{wij}=\sigma_{vij}=\sigma_{kij}=0.1, \quad \mu=1$$

All initial values are set to zeros. Sampling period is 10 ms. The desired trajectories of both the first link and the second link are obtained by passing a square wave signal of amplitude 5 and 20 s period into the filter $1/(s+2)^3$. Figure 10 shows experimental results. Both link angular positions θ_1 and θ_2 are able to follow their desired trajectories quite closely.

7 Conclusions

Our proposed model-independent control techniques—direct state-feedback backstepping intelligent control, indirect state-feedback backstepping intelligent control, and output-feedback backstepping intelligent control—are shown to be effective in controlling a system that is too complicated to be modeled accurately by physical laws. These control techniques have overcome the limitations of traditional adaptive control and offline-learning intelligent systems. Unlike traditional adaptive control, where structures of the unknown functions are required, the neural network has its own structure that has been proved to approximate any continuous functions with arbitrary accuracy causing it to be applicable to more extensive problems. Using a nonlinear observer, the control system can also be designed from the output signal.

By using online learning, our control system has shown to deliver a fast-enough response to be implemented successfully in a trajectory-tracking task of a robot manipulator. Experimental results have shown that these techniques can be applied to a complicated system, such as the two-link flexible-joint robot manipulator.

We consider a trajectory-tracking task of a two-link flexible-joint robot manipulator in the horizontal plane. The second motor

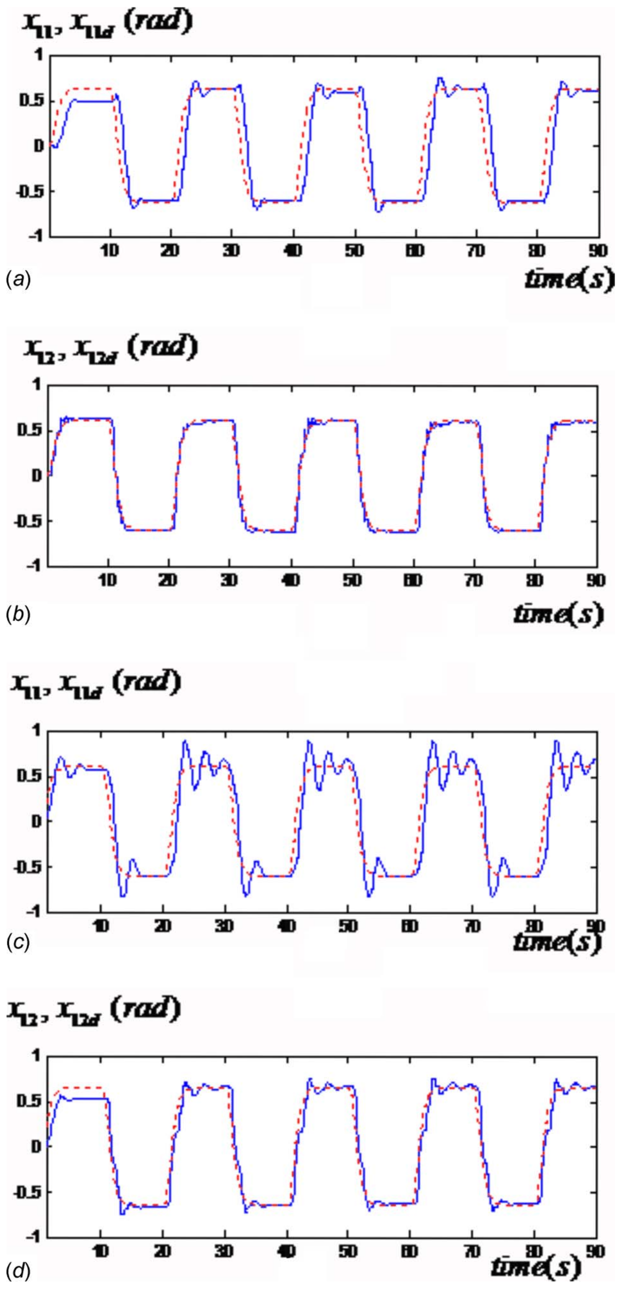

Fig. 10 Experimental results in 90 s. Indirect state-feedback control: (*a*) first link position and (*b*) second link position. Direct state-feedback control: (*c*) first link position and (*d*) second link position.

is attached to the first link, and its shaft does not share the same axis with the axis of rotation of the second link. This setting is different from the shared-axis cases commonly treated in the existing literature.

For state-feedback control, we are able to control the trajectory of the robot manipulator effectively using link angular position, link angular velocity, motor angular position, and motor angular velocity. For output-feedback control, we are able to control the trajectory of the robot manipulator effectively using only link angular position.

In the indirect control algorithm, the $\hat{g}_i^{-1}$ term can cause singularity. Some projection algorithms given in Ref. [20] can be used to avoid this problem. Note that the direct control algorithm does not have the inverse terms and hence avoids this problem.

Actuator nonlinearities, such as deadzone and backlash, are handled effectively by our control system. These actuator nonlinearities usually exist in practice, and controller implementation usually requires additional sensors to measure their magnitude. Our control algorithm only requires their magnitude to be bounded; we do not need additional sensors.

Since all the weights are adjusted on-line, the algorithm requires a certain level of computational power, which reflects the sampling rate. However, in our application, the computational requirement of the algorithm is not more than what is achievable by an industrial controller board.

References

[1] Sweet, L. M., and Good, M. C., 1984, "Re-Definition of the Robot Motion Control Problem: Effects of Plant Dynamics, Drive System Constraints, and User Requirements," *Proceedings of the 23rd IEEE Conference on Decision and Control*, Las Vegas, NV, pp. 724–731.
[2] Spong, M. W., 1987, "Modeling and Control of Elastic Joint Robots," ASME J. Dyn. Syst., Meas., Control, **109**(4), pp. 310–319.
[3] Nicosia, S., Tomei, P., and Tornambe, A., 1988, "A Nonlinear Observer for Elastic Robots," IEEE J. Rob. Autom., **4**(1), pp. 45–52.
[4] Ge, S. S., 1996, "Adaptive Control Design for Flexible Joint Manipulators," Automatica, **32**(2), pp. 273–278.
[5] Ge, S. S., Lee, T. H., and Harris, C. J., 1998, *Adaptive Neural Network Control of Robotic Manipulators*, World Scientific, Singapore.
[6] Spong, M., and Vidyasagar, M., 1989, *Robot Dynamics and Control*, Wiley, New York.
[7] De Luca, A., and Lucibello, P., 1998, "A General Algorithm for Dynamic Feedback Linearization of Robots With Elastic Joints," *Proceedings of the IEEE International Conference on Robotics and Automation*, Belgium, pp. 504–510.
[8] Brogliato, B., Ortega, R., and Lozano, R., 1995, "Global Tracking Controllers for Flexible-Joint Manipulators: A Comparative Study," Automatica, **31**(7), pp. 941–956.
[9] Huang, A. C., and Chen, Y. C., 2004, "Adaptive Sliding Control for Single-Link Flexible-Joint Robot With Mismatched Uncertainties," IEEE Trans. Control Syst. Technol., **12**(5), pp. 770–775.
[10] Park, C. W., 2004, "Robust Stable Fuzzy Control Via Fuzzy Modeling and Feedback Linearization With Its Applications to Controlling Uncertain Single-Link Flexible Joint Manipulators," J. Intell. Robotic Syst., **39**, pp. 131–147.
[11] Subudhi, B., and Morris, A. S., 2003, "Singular Perturbation Approach to Trajectory Tracking of Flexible Robot With Joint Elasticity," Int. J. Syst. Sci., **34**(3), pp. 167–179.
[12] Albu-Schäffer, A., and Hirzinger, G., 2001, "A Globally Stable State Feedback Controller for Flexible Joint Robots," Adv. Rob., **15**(8), pp. 799–814.
[13] Khalil, H. K., 2002, *Nonlinear Systems*, 3rd ed., Prentice Hall, New Jersey.
[14] Canudas de Wit, C., Olsson, H., Astrom, K. J., and Lischinsky, P., 1995, "A New Model for Control of Systems With Friction," IEEE Trans. Autom. Control, **40**(3), pp. 419–425.
[15] Tao, G., and Kokotovic, P. V., 1996, *Adaptive Control of Systems With Actuator and Sensor Nonlinearities*, Wiley, New York.
[16] Lewis, F. L., Campos, J., and Selmic, R., 2002, *Neuro-Fuzzy Control of Industrial Systems With Actuator Nonlinearities*, SIAM, Philadelphia.
[17] Ge, S. S., Hang, C. C., Lee, T. H., and Zhang, T., 2002, *Stable Adaptive Neural Network Control*, Kluwer, the Netherlands.
[18] Funahashi, K. I., 1989, "On the Approximate Realization of Continuous Mappings by Neural Networks," Neural Networks, **2**, pp. 183–192.
[19] Barron, A. R., 1993, "Universal Approximation Bounds for Superpositions of a Sigmoid Function," IEEE Trans. Inf. Theory, **39**(3), pp. 930–945.
[20] Chatlatanagulchai, W., 2006, "Backstepping Intelligent Control Applied to a Flexible-Joint Robot Manipulator," Ph.D. thesis, Department of Mechanical Engineering, Purdue University, West Lafayette, IN.