Adaptive Neural Network Control of Coordinated Manipulators

L.C. Woon, S.S. Ge*
Department of Electrical Engineering
National University of Singapore
Singapore 119260
e-mail: elegesz@nus.edu.sg

X.Q. Chen
Automation Division
Gintic Institute of Manufacturing Technology
Singapore 638075

C. Zhang
Department of Electrical and Electronic Engineering
University of Melbourne
Victoria 3052, Australia

Received February 13, 1998; accepted November 28, 1998

In this article, adaptive neural network control of coordinated manipulators is considered in an effort to eliminate the time-consuming and error prone dynamic modeling process which is necessary for the implementation of conventional adaptive control. After a concise dynamic model in the object coordinate space is developed for the coordinated manipulators, an adaptive neural network controller is presented by combining the techniques of neural network parameterization, adaptive control, and sliding mode control. It can be shown that the motion tracking errors converge to zero asymptotically whereas the internal force tracking error remains bounded and can be made arbitrarily small. Numerical simulations are conducted to show the effectiveness of the proposed method.

* To whom all correspondence should be addressed.

Journal of Robotic Systems 16(4), 195–211 (1999)

CCC 0741-2223/99/040195-17

1. INTRODUCTION

Coordinated control of multiple manipulators has attracted the attention of many researchers.[1–6] Interest in such systems stems from the greater capability of the manipulators in carrying out more complicated and dextrous tasks which may not be accomplished by a single manipulator. It is an important technology for applying manipulators to industrial automation, particularly in advanced flexible manufacturing systems.

The control of multiple manipulators presents a significant increase in complexity over the single manipulator case. The difficulties of the control problem lie in the fact that, when multiple manipulators grasp a common object, they form a closed kinematic chain mechanism. This will impose a set of kinematic and dynamic constraints on the position and velocity of the manipulators. As a result, the degrees of freedom of the whole system decrease, and internal forces are generated which need to be controlled.

A number of control methods have been proposed for solving this problem.[7–11] These methods can be classified into two categories as: (i) master–slave method,[7–9] where one or a group of robot manipulators play the role of the master, and the rest of the manipulators form the slave group which are moved in conjunction with the master manipulators; and (ii) hybrid position–force control method,[10,11] where the position of the object is controlled in a certain direction of the workspace, and the force is controlled in the other direction. Each robot manipulator is controlled using both position and force regulators.

It should be noted that the success of the aforementioned schemes for coordinated control of multiple manipulators relies on the full knowledge of the complex dynamics of the system. Care should therefore be taken if there is uncertainty about the system dynamics, as the controller so designed may give degraded performance and may incur instability. To deal with the uncertainties in the dynamics, several adaptive and robust coordinated control schemes have been proposed in refs. 1–6.

Recently, some developments have been made in the use of neural networks for the control of robot manipulators.[12–15] In general, neural network control design is done in two steps. First, a neural network is used to approximate the dynamic model of the system. This approximation is usually carried out off-line and then when a sufficiently accurate approximation is obtained, an appropriate control strategy using this approximation can be constructed. This approach has been shown to work well for many systems. However, it does not have any built-in capability to handle changes in the system. This is where incorporation of adaptive control is useful.

Some recent works have successfully incorporated adaptive techniques by using a suitable neural network to directly parameterize the control law.[16–22] Thus, the weights of the neural networks are updated on-line to eliminate the time-consuming off-line training process of the neural network. As a result, adaptive enhancements are introduced to the neural network controllers to improve their robustness to parameter uncertainties and model changes. Nonetheless, little work has been done to extend these techniques to coordinated control of multiple robot manipulators.

In this article, the neural network modeling technique and control method proposed in ref. 20 are applied to the control of multiple manipulators. The proposed method has the advantages that it does not require the time-consuming training process and the inverse of the dynamic system. The weights of the neural network can simply be initialized to zero by assuming no knowledge about the system. In addition, robust control can be easily incorporated into the control law to suppress the inherent neural network modeling errors and bounded disturbances. Because of the use of neural network parameterization techniques, the time-consuming and error prone dynamic modeling process is also eliminated, which is especially true for high degree-of-freedom robots.

This article is organized as follows. In section 2, the problem of neural network approximation, and the definitions for GL matrix and its product operator are briefly described. In section 3, the dynamics of coordinated manipulators are developed. In section 4, the adaptive neural network coordinated control scheme is presented and this is followed by a simulation study in section 5. Finally, some concluding remarks are stated in section 6.

2. NEURAL NETWORK APPROXIMATION

It has been shown that a neural network with a sufficiently large number of nodes can approximate any continuous function to any desired accuracy over a compact set. In control engineering, neural

network is commonly used as a function approximator that emulates some given nonlinear function $\mathscr{F}(\mathbf{y})$: $\mathbb{R}^n \to \mathbb{R}^m$ up to certain error tolerance ε. As a function emulator, first, an approximating function $\hat{\mathscr{F}}(\mathbf{W}, \mathbf{y})$: $\mathbb{R}^{m \times n_h} \times \mathbb{R}^n \to \mathbb{R}^m$ for $\mathscr{F}(\mathbf{y})$ is chosen, then an algorithm is used to update the parameters $\mathbf{W}$ based on the output error. In this article, neural networks are first used to parameterize the complex and unknown nonlinear dynamic functions of robots. Because of the use of neural network parameterization, the time-consuming and error prone dynamic modeling process is eliminated, so is the derivation of the linear in the parameter forms. Then, a stable adaptive control system is developed by utilizing the neural network parameterization model and direct adaptive control techniques. For the ease of derivation, the architecture of the neural network is chosen such that it can be linearly parameterized so that direct adaptive laws can be easily used to update the parameters of the neural network on-line. The radial basis function (RBF) neural network is most suitable for use in this case as the architecture belongs to the class of linearly parameterized models and it is possible to construct adaptive laws for updating the parameters of RBF neural network.

There exist many possible choices for the activation function, the Gaussian function is chosen following the works in ref. 20. Gaussian RBF network has been quite successful in representing the nonlinear function. It has been shown that a linear superposition of Gaussian radial basis functions results in an optimal mean square approximation to an unknown function which is infinitely differentiable and whose values are specified by a finite set of points in $\mathbb{R}^n$.[23] Furthermore, it has been proven that any continuous functions, not necessarily infinitely smooth, can be uniformly approximated by a linear combination of Gaussians.[24,25]

2.1. GL Matrix and Operator

To facilitate the analysis of neural networks, the GL matrix and its product operator introduced in refs. 20, 22, and 26 are briefly discussed here for completeness. Denote the GL vectors and matrices by $\{\cdot\}$ and the GL product operator by "$\bullet$." To avoid any possible confusion, $[\cdot]$ is used to denote the conventional vector and matrix.

In a general form, the GL matrix is a rectangular array of vectors, Let $\boldsymbol{\theta}_{kj} \in \mathbb{R}^{n_{kj}}$, where $k = 1, 2, \ldots, n$, $j = 1, 2, \ldots, m$. The GL matrix $\{\boldsymbol{\Theta}\}$ is defined in the following way,

$$\{\boldsymbol{\Theta}\} = \begin{Bmatrix} \boldsymbol{\theta}_{11} & \boldsymbol{\theta}_{12} & \cdots & \boldsymbol{\theta}_{1m} \\ \boldsymbol{\theta}_{21} & \boldsymbol{\theta}_{22} & \cdots & \boldsymbol{\theta}_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{\theta}_{n1} & \boldsymbol{\theta}_{n2} & \cdots & \boldsymbol{\theta}_{nm} \end{Bmatrix} \tag{1}$$

It should be noted that the dimension of each vector in a GL matrix may be different from each other. However, as long as n_{kj} is known, the structure of the GL matrix is uniquely defined. When $n = 1$ or $m = 1$, one would obtain the GL row vector or GL column vector, respectively. A conventional matrix (vector) can be taken as a GL matrix (vector) by partitioning the elements of the matrix (vector) into the subvector accordingly.

The transpose of a GL matrix $\{\boldsymbol{\Theta}\}$ is defined accordingly as

$$\{\boldsymbol{\Theta}\}^T = \begin{Bmatrix} \boldsymbol{\theta}_{11}^T & \boldsymbol{\theta}_{12}^T & \cdots & \boldsymbol{\theta}_{1m}^T \\ \boldsymbol{\theta}_{21}^T & \boldsymbol{\theta}_{22}^T & \cdots & \boldsymbol{\theta}_{2m}^T \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{\theta}_{n1}^T & \boldsymbol{\theta}_{n2}^T & \cdots & \boldsymbol{\theta}_{nm}^T \end{Bmatrix} \tag{2}$$

It can be seen that the transpose of a GL matrix is obtained by the transpose of its elementary vector locally.

Let $\{\boldsymbol{\Xi}\}$ be a given GL matrix defined as

$$\{\boldsymbol{\Xi}\} = \begin{Bmatrix} \boldsymbol{\xi}_{11} & \boldsymbol{\xi}_{12} & \cdots & \boldsymbol{\xi}_{1m} \\ \boldsymbol{\xi}_{21} & \boldsymbol{\xi}_{22} & \cdots & \boldsymbol{\xi}_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{\xi}_{n1} & \boldsymbol{\xi}_{n2} & \cdots & \boldsymbol{\xi}_{nm} \end{Bmatrix} \tag{3}$$

with $\boldsymbol{\xi}_{kj}$ having the same dimension as $\boldsymbol{\theta}_{kj}$, the GL product of $\{\boldsymbol{\Theta}\}^T$ and $\{\boldsymbol{\Xi}\}$ is an $n \times m$ matrix of elementwise products defined as

$$[\{\boldsymbol{\Theta}\}^T \bullet \{\boldsymbol{\Xi}\}] = \begin{bmatrix} \boldsymbol{\theta}_{11}^T \boldsymbol{\xi}_{11} & \boldsymbol{\theta}_{12}^T \boldsymbol{\xi}_{12} & \cdots & \boldsymbol{\theta}_{1m}^T \boldsymbol{\xi}_{1m} \\ \boldsymbol{\theta}_{21}^T \boldsymbol{\xi}_{21} & \boldsymbol{\theta}_{22}^T \boldsymbol{\xi}_{22} & \cdots & \boldsymbol{\theta}_{2m}^T \boldsymbol{\xi}_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{\theta}_{n1}^T \boldsymbol{\xi}_{n1} & \boldsymbol{\theta}_{n2}^T \boldsymbol{\xi}_{n2} & \cdots & \boldsymbol{\theta}_{nm}^T \boldsymbol{\xi}_{nm} \end{bmatrix} \tag{4}$$

Clearly, the GL product can be regarded as a generalization of the Hadamard matrix product.[27]

Note that in a mixed matrix product, the GL product should be computed first. For instance, in

{A}•{B}C, the matrix [{A}•{B}] should be computed first and then followed by the multiplication of [{A}•{B}] with matrix **C**.

3. DYNAMICS OF COORDINATED MANIPULATORS

3.1. System Description and Assumptions

Consider m manipulators holding a common object in a task space with n degrees of freedom. As shown in Figure 1, different coordinate frames have been established for system modeling. $OXYZ$ is the inertial reference frame in which the position and orientation of the manipulator end-effectors and object are referred. $O_o X_o Y_o Z_o$ is the object coordinate frame fixed at the center of mass of the object. $O_{ei} X_{ei} Y_{ei} Z_{ei}$ is the end-effector frame of the ith manipulator located at the grasp point.

To facilitate the dynamic formulation, the following assumptions are made.[2,4]

Assumption 3.1: *Each manipulator is nonredundant, i.e., the number of degrees of freedom of each manipulator is equal to the dimension of task space.*

Assumption 3.2: *All the end-effectors of the manipulators are rigidly attached to the common object so that there is no relative motion between the object and the end-effectors.*

Assumption 3.3: *The kinematics for each manipulator are known exactly, and each manipulator is moved such that kinematics singularities are avoided. This implies that the Jacobian matrix of each manipulator is known and has full rank.*

Assumption 3.4: *The object is rigid, that is, the object does not get deformed with the application of forces.*

Remark 3.1: It should be noted that if rolling and sliding contacts exist between the object and any end-effector, the problem will become even more complicated, due to the existence of nonholonomic constraint. This problem, however, is beyond the scope of this article. The works in refs. 28 and 29 may provide some insight on the subject.

3.2. Dynamics of Manipulators

The dynamic equation of the ith manipulator in joint space is given by[30,31]

$$\mathbf{D}_i(\mathbf{q}_i)\ddot{\mathbf{q}}_i + \mathbf{C}_i(\mathbf{q}_i, \dot{\mathbf{q}}_i)\dot{\mathbf{q}}_i + \mathbf{G}_i(\mathbf{q}_i) + \mathbf{J}_{ei}^T(\mathbf{q}_i)\mathbf{F}_{ei} = \boldsymbol{\tau}_i \qquad i = 1, 2, \ldots, m \quad (5)$$

where $\mathbf{q}_i \in \mathbb{R}^n$ is the vector of joint variables, $\boldsymbol{\tau}_i \in \mathbb{R}^n$ is the vector of joint torque applied by the actuators, $\mathbf{D}_i(\mathbf{q}_i) \in \mathbb{R}^{n\times n}$ is the symmetric positive definite inertia matrix, $\mathbf{C}_i(\mathbf{q}_i, \dot{\mathbf{q}}_i) \in \mathbb{R}^{n\times n}$ is the Coriolis and centrifugal matrix, $\mathbf{G}_i(\mathbf{q}_i) \in \mathbb{R}^n$ is the vector of gravitational forces, $\mathbf{F}_{ei} \in \mathbb{R}^n$ is the force vector exerted by the ith end-effector on the object at the grasp point, and $\mathbf{J}_{ei}(\mathbf{q}_i) \in \mathbb{R}^{n\times n}$ is the manipulator Jacobian matrix.

The dynamic Eq. (5) for the m manipulators can be expressed concisely as

$$\mathbf{D}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) + \mathbf{J}_e^T(\mathbf{q})\mathbf{F}_e = \boldsymbol{\tau} \quad (6)$$

where

$$\mathbf{q} = \begin{bmatrix} \mathbf{q}_1^T & \mathbf{q}_2^T & \cdots & \mathbf{q}_m^T \end{bmatrix}^T \in \mathbb{R}^{mn}$$

$$\boldsymbol{\tau} = \begin{bmatrix} \boldsymbol{\tau}_1^T & \boldsymbol{\tau}_2^T & \cdots & \boldsymbol{\tau}_m^T \end{bmatrix}^T \in \mathbb{R}^{mn}$$

$$\mathbf{D}(\mathbf{q}) = \text{block diag}\begin{bmatrix} \mathbf{D}_1(\mathbf{q}_1) & \mathbf{D}_2(\mathbf{q}_2) & \cdots & \mathbf{D}_m(\mathbf{q}_m) \end{bmatrix} \in \mathbb{R}^{mn\times mn}$$

$$\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) = \text{block diag}\begin{bmatrix} \mathbf{C}_1(\mathbf{q}_1, \dot{\mathbf{q}}_1) & \mathbf{C}_2(\mathbf{q}_2, \dot{\mathbf{q}}_2) & \cdots & \mathbf{C}_m(\mathbf{q}_m, \dot{\mathbf{q}}_m) \end{bmatrix} \in \mathbb{R}^{mn\times mn}$$

$$\mathbf{G}(\mathbf{q}) = \begin{bmatrix} \mathbf{G}_1^T(\mathbf{q}_1) & \mathbf{G}_2^T(\mathbf{q}_2) & \cdots & \mathbf{G}_m^T(\mathbf{q}_m) \end{bmatrix}^T \in \mathbb{R}^{mn}$$

$$\mathbf{F}_e = \begin{bmatrix} \mathbf{F}_{e1}^T & \mathbf{F}_{e2}^T & \cdots & \mathbf{F}_{em}^T \end{bmatrix}^T \in \mathbb{R}^{mn}$$

$$\mathbf{J}_e(\mathbf{q}) = \text{block diag}\begin{bmatrix} \mathbf{J}_{e1}(\mathbf{q}_1) & \mathbf{J}_{e2}(\mathbf{q}_2) & \cdots & \mathbf{J}_{em}(\mathbf{q}_m) \end{bmatrix} \in \mathbb{R}^{mn\times mn}$$

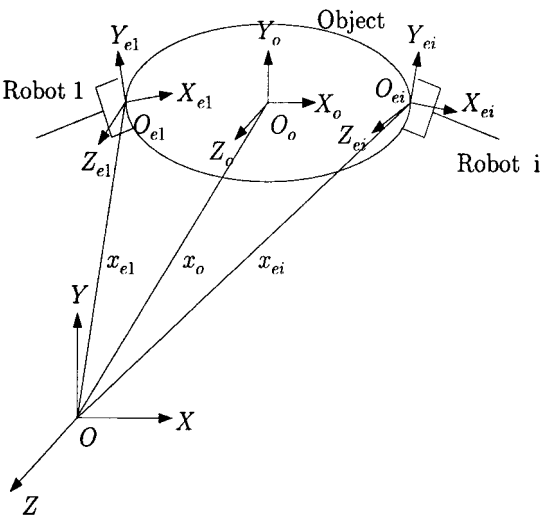

Figure 1. Coordinate system.

3.3. Object Dynamics

The equation of motion of the object is described by[2,3]

$$\mathbf{D}_o(\mathbf{x}_o)\ddot{\mathbf{x}}_o + \mathbf{C}_o(\mathbf{x}_o, \dot{\mathbf{x}}_o)\dot{\mathbf{x}}_o + \mathbf{G}_o(\mathbf{x}_o) = \mathbf{F}_o \qquad (7)$$

where $\mathbf{x}_o \in \mathbb{R}^n$ is the position and orientation vector of the object frame, $\mathbf{F}_o \in \mathbb{R}^n$ is the resultant force acting on the center of mass of the object, $\mathbf{D}_o(\mathbf{x}_o) \in \mathbb{R}^{n\times n}$ is the symmetric positive definite inertial matrix of the object, $\mathbf{C}_o(\mathbf{x}_o, \dot{\mathbf{x}}_o) \in \mathbb{R}^{n\times n}$ is the Coriolis and centrifugal matrix, and $\mathbf{G}_o(\mathbf{x}_o) \in \mathbb{R}^n$ is the gravitational force vector.

Define $\mathbf{J}_o(\mathbf{x}_o) \in \mathbb{R}^{mn\times n}$ as

$$\mathbf{J}_o(\mathbf{x}_o) = \begin{bmatrix} \mathbf{J}_{o1}^T(\mathbf{x}_o) & \mathbf{J}_{o2}^T(\mathbf{x}_o) & \cdots & \mathbf{J}_{om}^T(\mathbf{x}_o) \end{bmatrix}^T$$

where $\mathbf{J}_{oi}(\mathbf{x}_o) \in \mathbb{R}^{n\times n}$ is the Jacobian matrix from the object frame $O_o X_o Y_o Z_o$ to the ith manipulator end-effector frame $O_{ei} X_{ei} Y_{ei} Z_{ei}$. Then $\mathbf{F}_o$ is given by[2]

$$\mathbf{F}_o = \mathbf{J}_o^T(\mathbf{x}_o)\mathbf{F}_e \qquad (8)$$

Given the resultant force $\mathbf{F}_o$, the end-effector force $\mathbf{F}_e$ that satisfies (8) can be decomposed into two orthogonal components, one that contributes to the motion of the object, and one that produces the internal force. This is clearly represented by the following equation,[2,4]

$$\mathbf{F}_e = (\mathbf{J}_o^T(\mathbf{x}_o))^+ \mathbf{F}_o + \mathbf{F}_I \qquad (9)$$

where $(\mathbf{J}_o^T(\mathbf{x}_o))^+ \in \mathbb{R}^{mn\times n}$ is the pseudo-inverse matrix of $\mathbf{J}_o^T(\mathbf{x}_o)$, and $\mathbf{F}_I \in \mathbb{R}^{mn}$ is the internal force vector, which is in the null space of $\mathbf{J}_o^T(\mathbf{x}_o)$, i.e., satisfying

$$\mathbf{J}_o^T(\mathbf{x}_o)\mathbf{F}_I = 0$$

Substituting for $\mathbf{F}_o$ from (7) into (9), results in

$$\mathbf{F}_e = (\mathbf{J}_o^T(\mathbf{x}_o))^+ (\mathbf{D}_o(\mathbf{x}_o)\ddot{\mathbf{x}}_o + \mathbf{C}_o(\mathbf{x}_o, \dot{\mathbf{x}}_o)\dot{\mathbf{x}}_o + \mathbf{G}_o(\mathbf{x}_o)) + \mathbf{F}_I \qquad (10)$$

3.4. Combined Dynamics

It is worth noting that there are in total $(m+1)n$ position variables in (6) and (7), but in fact, only n of them are independent. This can be easily seen since once the trajectory of the object is given, the joint trajectory of each robot is uniquely determined due to Assumptions 3.1 and 3.2. Owing to the dependence among position variables, (6) and (7) are not suitable for one to analyze the dynamic behavior of the system. With this observation in mind, the object position $\mathbf{x}_o$ will be treated as independent position variables and the dynamic equation (6) will be reformulated in terms of $\mathbf{x}_o$ so as to describe the whole system behavior.

Let $\mathbf{x}_{ei} \in \mathbb{R}^n$ denote the position and orientation vector of the ith end-effector. Then $\dot{\mathbf{x}}_{ei}$ is related to $\dot{\mathbf{q}}_i$ by the Jacobian matrix $\mathbf{J}_{ei}(\mathbf{q}_i)$ as

$$\dot{\mathbf{x}}_{ei} = \mathbf{J}_{ei}(\mathbf{q}_i)\dot{\mathbf{q}}_i \qquad (11)$$

and the relationship between $\dot{\mathbf{x}}_{ei}$ and $\dot{\mathbf{x}}_o$ is given by

$$\dot{\mathbf{x}}_{ei} = \mathbf{J}_{oi}(\mathbf{x}_o)\dot{\mathbf{x}}_o \qquad (12)$$

After combining (11) and (12), the following relationship between the joint velocity of the ith manipulator and the velocity of the object is obtained

$$\mathbf{J}_{ei}(\mathbf{q}_i)\dot{\mathbf{q}}_i = \mathbf{J}_{oi}(\mathbf{x}_o)\dot{\mathbf{x}}_o \qquad (13)$$

As it is assumed that the manipulators work in a nonsingular region, thus the inverse of the Jacobian matrix $\mathbf{J}_{ei}(\mathbf{q}_i)$ exists. Considering all the manipulators acting on the object at the same time, yields

$$\dot{\mathbf{q}} = \mathbf{J}_e^{-1}(\mathbf{q})\mathbf{J}_o(\mathbf{x}_o)\dot{\mathbf{x}}_o \qquad (14)$$

Differentiating (14) with respect to time leads to

$$\ddot{\mathbf{q}} = \mathbf{J}_e^{-1}(\mathbf{q})\mathbf{J}_o(\mathbf{x}_o)\ddot{\mathbf{x}}_o + \frac{d}{dt}(\mathbf{J}_e^{-1}(\mathbf{q})\mathbf{J}_o(\mathbf{x}_o))\dot{\mathbf{x}}_o \quad (15)$$

Using (10), (14), and (15), the dynamic equation (6) can be written as

$$\begin{aligned}&\mathbf{D}(\mathbf{q})\mathbf{J}_e^{-1}(\mathbf{q})\mathbf{J}_o(\mathbf{x}_o)\ddot{\mathbf{x}}_o + \mathbf{D}(\mathbf{q})\frac{d}{dt}(\mathbf{J}_e^{-1}(\mathbf{q})\mathbf{J}_o(\mathbf{x}_o))\dot{\mathbf{x}}_o\\ &\quad + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\mathbf{J}_e^{-1}(\mathbf{q})\mathbf{J}_o(\mathbf{x}_o)\dot{\mathbf{x}}_o\\ &\quad + \mathbf{G}(\mathbf{q}) + \mathbf{J}_e^T(\mathbf{q})(\mathbf{J}_o^T(\mathbf{x}_o))^+\\ &\quad \times(\mathbf{D}_o(\mathbf{x}_o)\ddot{\mathbf{x}}_o + \mathbf{C}_o(\mathbf{x}_o,\dot{\mathbf{x}}_o)\dot{\mathbf{x}}_o + \mathbf{G}_o(\mathbf{x}_o)) + \mathbf{J}_e^T(\mathbf{q})\mathbf{F}_I\\ &\quad = \boldsymbol{\tau} \qquad (16)\end{aligned}$$

Premultiplying both sides of the above equation by $\mathbf{J}_o^T(\mathbf{x}_o)\mathbf{J}_e^{-T}(\mathbf{q})$, and using the fact that $\mathbf{J}_o^T(\mathbf{x}_o)\mathbf{F}_I = 0$, the dynamic model of the multiple manipulators system (6), coupled with the object dynamics (7), is then given by

$$\mathbf{D}_M(\mathbf{x}_o)\ddot{\mathbf{x}}_o + \mathbf{C}_M(\mathbf{x}_o,\dot{\mathbf{x}}_o)\dot{\mathbf{x}}_o + \mathbf{G}_M(\mathbf{x}_o) = \mathbf{u}_M \quad (17)$$

where

$$\begin{aligned}\mathbf{D}_M(\mathbf{x}_o) &= \mathbf{J}_o^T(\mathbf{x}_o)\mathbf{J}_e^{-T}(\mathbf{q})\mathbf{D}(\mathbf{q})\mathbf{J}_e^{-1}(\mathbf{q})\\ &\quad \mathbf{J}_o(\mathbf{x}_o) + \mathbf{D}_o(\mathbf{x}_o) \qquad (18)\\ \mathbf{C}_M(\mathbf{x}_o,\dot{\mathbf{x}}_o) &= \mathbf{J}_o^T(\mathbf{x}_o)\mathbf{J}_e^{-T}(\mathbf{q})\mathbf{D}(\mathbf{q})\\ &\quad \frac{d}{dt}(\mathbf{J}_e^{-1}(\mathbf{q})\mathbf{J}_o(\mathbf{x}_o))\\ &\quad + \mathbf{J}_o^T(\mathbf{x}_o)\mathbf{J}_e^{-T}(\mathbf{q})\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\mathbf{J}_e^{-1}(\mathbf{q})\\ &\quad \mathbf{J}_o(\mathbf{x}_o) + \mathbf{C}_o(\mathbf{x}_o,\dot{\mathbf{x}}_o) \qquad (19)\\ \mathbf{G}_M(\mathbf{x}_o) &= \mathbf{J}_o^T(\mathbf{x}_o)\mathbf{J}_e^{-T}(\mathbf{q})\mathbf{G}(\mathbf{q}) + \mathbf{G}_o(\mathbf{x}_o) \qquad (20)\\ \mathbf{u}_M &= \mathbf{J}_o^T(\mathbf{x}_o)\mathbf{J}_e^{-T}(\mathbf{q})\boldsymbol{\tau} \qquad (21)\end{aligned}$$

The dynamic equation (17) has the following structural properties, which can be exploited to facilitate the control system design.[1,4,6]

Property 3.1: *The matrix* $\mathbf{D}_M(\mathbf{x}_o)$ *is symmetric, positive definite, and is bounded from below and above, i.e.,*

$$\lambda_{\min} I \le \mathbf{D}_M(\mathbf{x}_o) \le \lambda_{\max} I$$

where λ_{min} *and* $\lambda_{max} \in \mathbb{R}$ *denote the minimum and maximum eigenvalues of* $\mathbf{D}_M(\mathbf{x}_o)$.

Property 3.2: *The matrix* $\dot{\mathbf{D}}_M(\mathbf{x}_o) - 2\mathbf{C}_M(\mathbf{x}_o,\dot{\mathbf{x}}_o)$ *is skew-symmetric, that is,*

$$\mathbf{r}^T\left(\dot{\mathbf{D}}_M(\mathbf{x}_o) - 2\mathbf{C}_M(\mathbf{x}_o,\dot{\mathbf{x}}_o)\right)\mathbf{r} = 0$$

for any vector $\mathbf{r} \in \mathbb{R}^n$.

Property 3.3: *The left-hand side of* (17) *can be written in the linear in the parameters form as follows*

$$\begin{aligned}&\mathbf{D}_M(\mathbf{x}_o)\ddot{\mathbf{x}}_o + \mathbf{C}_M(\mathbf{x}_o,\dot{\mathbf{x}}_o)\dot{\mathbf{x}}_o + \mathbf{G}_M(\mathbf{x}_o)\\ &\quad = \mathbf{Y}_M(\mathbf{x}_o,\dot{\mathbf{x}}_o,\ddot{\mathbf{x}}_o)\boldsymbol{\varphi}_M\end{aligned}$$

where $\mathbf{Y}_M(\mathbf{x}_o,\dot{\mathbf{x}}_o,\ddot{\mathbf{x}}_o) \in \mathbb{R}^{n\times n_p}$ *is a matrix of known function, known as the regressor matrix, and* $\boldsymbol{\varphi}_M \in \mathbb{R}^{n_p}$ *is a vector of parameters.*

4. CONTROLLER FORMULATION

4.1. Neural Network Based Modeling

To eliminate the time-consuming and error prone dynamic modeling process, neural network based modeling approach is considered in this article. Since $\mathbf{D}_M(\mathbf{x}_o)$ and $\mathbf{G}_M(\mathbf{x}_o)$ are functions of $\mathbf{x}_o$ only, neural networks with inputs $\mathbf{x}_o$ are sufficient to model them.[20,26] Define the neural network functional approximations for $d_{Mkj}(\mathbf{x}_o)$ and $g_{Mk}(\mathbf{x}_o)$ as follows,

$$\begin{aligned}d_{Mkj}(\mathbf{x}_o) &= \boldsymbol{\theta}_{\mathbf{D}kj}^T\boldsymbol{\xi}_{\mathbf{D}kj}(\mathbf{x}_o) + \varepsilon_{\mathbf{D}kj}(\mathbf{x}_o)\\ g_{Mk}(\mathbf{x}_o) &= \boldsymbol{\theta}_{\mathbf{G}k}^T\boldsymbol{\xi}_{\mathbf{G}k}(\mathbf{x}_o) + \varepsilon_{\mathbf{G}k}(\mathbf{x}_o)\end{aligned}$$

where $\boldsymbol{\theta}_{\mathbf{D}kj}, \boldsymbol{\theta}_{\mathbf{G}k}$ are the vectors of neural network weights, $\xi_{\mathbf{D}kj}(\mathbf{x}_o)$, $\xi_{\mathbf{G}k}(\mathbf{x}_o)$ are the vectors of activation functions, and $\varepsilon_{\mathbf{D}kj}(\mathbf{x}_o), \varepsilon_{\mathbf{G}k}(\mathbf{x}_o) \in \mathbb{R}$ are the neural network reconstruction errors (also referred to as ''approximation errors'' or ''modeling errors'') of $d_{Mkj}(\mathbf{x}_o)$ and $g_{Mk}(\mathbf{x}_o)$, respectively, which are assumed to be bounded.

On the other hand, $\mathbf{C}_M(\mathbf{x}_o,\dot{\mathbf{x}}_o)$ is a matrix of $\mathbf{x}_o$ and $\dot{\mathbf{x}}_o$, neural networks with inputs $\mathbf{x}_o$ and $\dot{\mathbf{x}}_o$ are needed to model it.[20,26] Assume that $c_{Mkj}(\mathbf{x}_o,\dot{\mathbf{x}}_o)$ can be approximated as

$$c_{Mkj}(\mathbf{x}_o,\dot{\mathbf{x}}_o) = \boldsymbol{\theta}_{\mathbf{C}kj}^T\boldsymbol{\xi}_{\mathbf{C}kj}(\mathbf{x}_o,\dot{\mathbf{x}}_o) + \varepsilon_{\mathbf{C}kj}(\mathbf{x}_o,\dot{\mathbf{x}}_o)$$

where $\boldsymbol{\theta}_{\mathbf{C}kj}$ is the vector of neural network weights, $\boldsymbol{\xi}_{\mathbf{C}kj}(\mathbf{x}_o,\dot{\mathbf{x}}_o)$ is the corresponding vector of activation function, and $\varepsilon_{\mathbf{C}kj}(\mathbf{x}_o,\dot{\mathbf{x}}_o) \in \mathbb{R}$ is the reconstruction error of $c_{Mkj}(\mathbf{x}_o,\dot{\mathbf{x}}_o)$, which is also assumed to be bounded.

Thus, $\mathbf{D}_M(\mathbf{x}_o)$, $\mathbf{C}_M(\mathbf{x}_o, \dot{\mathbf{x}}_o)$, and $\mathbf{G}_M(\mathbf{x}_o)$ can be conveniently expressed as follows,

$$\mathbf{D}_M(\mathbf{x}_o) = \begin{bmatrix} \boldsymbol{\theta}_{\mathbf{D}11}^T \boldsymbol{\xi}_{\mathbf{D}11}(\mathbf{x}_o) & \boldsymbol{\theta}_{\mathbf{D}12}^T \boldsymbol{\xi}_{\mathbf{D}12}(\mathbf{x}_o) & \cdots & \boldsymbol{\theta}_{\mathbf{D}1n}^T \boldsymbol{\xi}_{\mathbf{D}1n}(\mathbf{x}_o) \\ \boldsymbol{\theta}_{\mathbf{D}21}^T \boldsymbol{\xi}_{\mathbf{D}21}(\mathbf{x}_o) & \boldsymbol{\theta}_{\mathbf{D}22}^T \boldsymbol{\xi}_{\mathbf{D}22}(\mathbf{x}_o) & \cdots & \boldsymbol{\theta}_{\mathbf{D}2n}^T \boldsymbol{\xi}_{\mathbf{D}2n}(\mathbf{x}_o) \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{\theta}_{\mathbf{D}n1}^T \boldsymbol{\xi}_{\mathbf{D}n1}(x) & \boldsymbol{\theta}_{\mathbf{D}n2}^T \boldsymbol{\xi}_{\mathbf{D}n2}(\mathbf{x}_o) & \cdots & \boldsymbol{\theta}_{\mathbf{D}nn}^T \boldsymbol{\xi}_{\mathbf{D}nn}(\mathbf{x}_o) \end{bmatrix} + \mathbf{E}_{\mathbf{D}}(\mathbf{x}_o)$$

$$\mathbf{C}_M(\mathbf{x}_o, \dot{\mathbf{x}}_o) = \begin{bmatrix} \boldsymbol{\theta}_{\mathbf{C}11}^T \boldsymbol{\xi}_{\mathbf{C}11}(\mathbf{x}_o, \dot{\mathbf{x}}_o) & \boldsymbol{\theta}_{\mathbf{C}12}^T \boldsymbol{\xi}_{\mathbf{C}12}(\mathbf{x}_o, \dot{\mathbf{x}}_o) & \cdots & \boldsymbol{\theta}_{\mathbf{C}1n}^T \boldsymbol{\xi}_{\mathbf{C}1n}(\mathbf{x}_o, \dot{\mathbf{x}}_o) \\ \boldsymbol{\theta}_{\mathbf{C}21}^T \boldsymbol{\xi}_{\mathbf{C}21}(\mathbf{x}_o, \dot{\mathbf{x}}_o) & \boldsymbol{\theta}_{\mathbf{C}22}^T \boldsymbol{\xi}_{\mathbf{C}22}(\mathbf{x}_o, \dot{\mathbf{x}}_o) & \cdots & \boldsymbol{\theta}_{\mathbf{C}2n}^T \boldsymbol{\xi}_{\mathbf{C}2n}(\mathbf{x}_o, \dot{\mathbf{x}}_o) \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{\theta}_{\mathbf{C}n1}^T \boldsymbol{\xi}_{\mathbf{C}n1}(\mathbf{x}_o, \dot{\mathbf{x}}_o) & \boldsymbol{\theta}_{\mathbf{C}n2}^T \boldsymbol{\xi}_{\mathbf{C}n2}(\mathbf{x}_o, \dot{\mathbf{x}}_o) & \cdots & \boldsymbol{\theta}_{\mathbf{C}nn}^T \boldsymbol{\xi}_{\mathbf{C}nn}(\mathbf{x}_o, \dot{\mathbf{x}}_o) \end{bmatrix} + \mathbf{E}_{\mathbf{C}}(\mathbf{x}_o, \dot{\mathbf{x}}_o)$$

$$\mathbf{G}_M(\mathbf{x}_o) = \begin{bmatrix} \boldsymbol{\theta}_{\mathbf{G}1}^T \boldsymbol{\xi}_{\mathbf{G}1}(\mathbf{x}_o) \\ \boldsymbol{\theta}_{\mathbf{G}2}^T \boldsymbol{\xi}_{\mathbf{G}2}(\mathbf{x}_o) \\ \vdots \\ \boldsymbol{\theta}_{\mathbf{G}n}^T \boldsymbol{\xi}_{\mathbf{G}n}(\mathbf{x}_o) \end{bmatrix} + \mathbf{E}_{\mathbf{G}}(\mathbf{x}_o)$$

where $\mathbf{E}_{\mathbf{D}}(\mathbf{x}_o)$, $\mathbf{E}_{\mathbf{C}}(\mathbf{x}_o, \dot{\mathbf{x}}_o) \in \mathbb{R}^{n \times n}$, and $\mathbf{E}_{\mathbf{G}}(\mathbf{x}_o) \in \mathbb{R}^n$ are the neural network reconstruction errors matrices and vector with their elements being the $\varepsilon_{\mathbf{D}kj}(\mathbf{x}_o)$, $\varepsilon_{\mathbf{C}kj}(\mathbf{x}_o, \dot{\mathbf{x}}_o)$, and $\varepsilon_{\mathbf{G}k}(\mathbf{x}_o)$, respectively.

Using the GL matrix and its product operator, the neural network approximations for $\mathbf{D}_M(\mathbf{x}_o)$, $\mathbf{C}_M(\mathbf{x}_o, \dot{\mathbf{x}}_o)$, and $\mathbf{G}_M(\mathbf{x}_o)$ can be further written as follows,

$$\mathbf{D}_M(\mathbf{x}_o) = \left[\{\boldsymbol{\Theta}_{\mathbf{D}}\}^T \bullet \{\boldsymbol{\Xi}_{\mathbf{D}}(\mathbf{x}_o)\}\right] + \mathbf{E}_{\mathbf{D}}(\mathbf{x}_o) \tag{22}$$

$$\mathbf{C}_M(\mathbf{x}_o, \dot{\mathbf{x}}_o) = \left[\{\boldsymbol{\Theta}_{\mathbf{C}}\}^T \bullet \{\boldsymbol{\Xi}_{\mathbf{C}}(\mathbf{x}_o, \dot{\mathbf{x}}_o)\}\right] + \mathbf{E}_{\mathbf{C}}(\mathbf{x}_o, \dot{\mathbf{x}}_o) \tag{23}$$

$$\mathbf{G}_M(\mathbf{x}_o) = \left[\{\boldsymbol{\Theta}_{\mathbf{G}}\}^T \bullet \{\boldsymbol{\Xi}_{\mathbf{G}}(\mathbf{x}_o)\}\right] + \mathbf{E}_{\mathbf{G}}(\mathbf{x}_o) \tag{24}$$

where $\{\boldsymbol{\Theta}_{\mathbf{D}}\}$, $\{\boldsymbol{\Theta}_{\mathbf{C}}\}$, and $\{\boldsymbol{\Theta}_{\mathbf{G}}\}$ are the GL matrices and vector with their elements being the neural network weights $\boldsymbol{\theta}_{\mathbf{D}kj}$, $\boldsymbol{\theta}_{\mathbf{C}kj}$, and $\boldsymbol{\theta}_{\mathbf{G}k}$, respectively; $\{\boldsymbol{\Xi}_{\mathbf{D}}(\mathbf{x}_o)\}$, $\{\boldsymbol{\Xi}_{\mathbf{C}}(\mathbf{x}_o, \dot{\mathbf{x}}_o)\}$, and $\{\boldsymbol{\Xi}_{\mathbf{G}}(\mathbf{x}_o)\}$ are the GL matrices and vector with their elements being the neural network activation functions $\boldsymbol{\xi}_{\mathbf{D}kj}(\mathbf{x}_o)$, $\boldsymbol{\xi}_{\mathbf{C}kj}(\mathbf{x}_o, \dot{\mathbf{x}}_o)$, and $\boldsymbol{\xi}_{\mathbf{G}k}(\mathbf{x}_o)$, respectively; $\mathbf{E}_{\mathbf{D}}(\mathbf{x}_o)$, $\mathbf{E}_{\mathbf{C}}(\mathbf{x}_o, \dot{\mathbf{x}}_o) \in \mathbb{R}^{n \times n}$, and $\mathbf{E}_{\mathbf{G}}(\mathbf{x}_o) \in \mathbb{R}^n$ are the neural network reconstruction error matrices and vector with their elements being the $\varepsilon_{\mathbf{D}kj}(\mathbf{x}_o)$, $\varepsilon_{\mathbf{C}kj}(\mathbf{x}_o, \dot{\mathbf{x}}_o)$, and $\varepsilon_{\mathbf{G}k}(\mathbf{x}_o)$, respectively.

4.2. Controller Design

Let $\mathbf{x}_{\mathrm{od}}(t) \in \mathbb{R}^n$ be the desired trajectory of the object and $\mathbf{F}_{\mathrm{Id}} \in \mathbb{R}^{mn}$ be the desired internal force which is chosen so that it is in the null space of $\mathbf{J}_o^T(\mathbf{x}_o)$. The control objective is to achieve asymptotic tracking of both the object position and the internal force. Define the tracking errors for the object position and the internal force as

$$\mathbf{e}_o(t) = \mathbf{x}_{\mathrm{od}}(t) - \mathbf{x}_o(t) \tag{25}$$

$$\mathbf{e}_I(t) = \mathbf{F}_{\mathrm{Id}}(t) - \mathbf{F}_I(t) \tag{26}$$

Define the reference velocity and filtered tracking errors as

$$\dot{\mathbf{x}}_{\mathrm{or}}(t) = \dot{\mathbf{x}}_{\mathrm{od}}(t) + \boldsymbol{\Lambda}\,\mathbf{e}_o(t) \tag{27}$$

$$\mathbf{r}_o(t) = \dot{\mathbf{x}}_{\mathrm{or}}(t) - \dot{\mathbf{x}}_o(t) = \dot{\mathbf{e}}_o(t) + \boldsymbol{\Lambda}\,\mathbf{e}_o(t) \tag{28}$$

where $\boldsymbol{\Lambda} \in \mathbb{R}^{n \times n}$ is a symmetric positive definite matrix. The signal $\dot{\mathbf{x}}_{\mathrm{or}}$ so defined guarantees that $\ddot{\mathbf{x}}_{\mathrm{or}}$ is only a function of $\dot{\mathbf{x}}_{\mathrm{od}}$ and $\dot{\mathbf{e}}_o$ and does not depend on the acceleration $\ddot{\mathbf{x}}_o$, which is difficult to measure in practice. With the help of the following lemma, the stability of $\mathbf{e}_o$ and $\dot{\mathbf{e}}_o$ can be concluded by studying $\mathbf{r}$.

Lemma 4.1: *Let* $\mathbf{e}_o(t) = \mathbf{h}(t) * \mathbf{r}_o(t)$, *where* "$*$" *denotes convolution product and* $\mathbf{h}(t) = L^{-1}(\mathbf{H}(s))$ *with* $\mathbf{H}(s)$ *is an* $(n-m) \times (n-m)$ *strictly proper, exponen-*

tially stable transfer function. Then $\mathbf{r}_o \in L_2^{n-m}$ *implies that* $\mathbf{e}_o \in L_2^{n-m} \cap L_\infty^{n-m}$, $\dot{\mathbf{e}}_o \in L_2^{n-m}$, $\mathbf{e}_o$ *is continuous, and* $\mathbf{e}_o \to 0$ *as* $t \to \infty$. *If, in addition,* $\mathbf{r}_o \to 0$ *as* $t \to \infty$, *then* $\dot{\mathbf{e}}_o \to 0$.[32]

Using (22)–(24) and (28), the dynamic equation given in (17) can be written as

$$\begin{aligned}\left[\{\boldsymbol{\Theta}_\mathbf{D}\}^T \bullet \{\boldsymbol{\Xi}_\mathbf{D}(\mathbf{x}_o)\}\right]\ddot{\mathbf{x}}_{\mathrm{or}} + \left[\{\boldsymbol{\Theta}_\mathbf{C}\}^T \bullet \{\boldsymbol{\Xi}_\mathbf{C}(\mathbf{x}_o, \dot{\mathbf{x}}_o)\}\right]\dot{\mathbf{x}}_{\mathrm{or}} \\ + \left[\{\boldsymbol{\Theta}_\mathbf{G}\}^T \bullet \{\boldsymbol{\Xi}_\mathbf{G}(\mathbf{x}_o)\}\right] \\ + \mathbf{E}(\mathbf{x}_o, \dot{\mathbf{x}}_o, \dot{\mathbf{x}}_{\mathrm{or}}, \ddot{\mathbf{x}}_{\mathrm{or}}) - \mathbf{D}_M(\mathbf{x}_o)\dot{\mathbf{r}}_o - \mathbf{C}_M(\mathbf{x}_o, \dot{\mathbf{x}}_o)\mathbf{r}_o \\ = \mathbf{u}_M\end{aligned}$$

where

$$\begin{aligned}&\mathbf{E}(\mathbf{x}_o, \dot{\mathbf{x}}_o, \dot{\mathbf{x}}_{\mathrm{or}}, \ddot{\mathbf{x}}_{\mathrm{or}}) \\ &\quad = \mathbf{E}_\mathbf{D}(\mathbf{x}_o)\ddot{\mathbf{x}}_{\mathrm{or}} + \mathbf{E}_\mathbf{C}(\mathbf{x}_o, \dot{\mathbf{x}}_o)\dot{\mathbf{x}}_{\mathrm{or}} + \mathbf{E}_\mathbf{G}(\mathbf{x}_o)\end{aligned}$$

Let $\hat{\mathbf{D}}_M(\mathbf{x}_o)$, $\hat{\mathbf{C}}_M(\mathbf{x}_o, \dot{\mathbf{x}}_o)$, and $\hat{\mathbf{G}}_M(\mathbf{x}_o)$ be the estimates of $\mathbf{D}_M(\mathbf{x}_o)$, $\mathbf{C}_M(\mathbf{x}_o, \dot{\mathbf{x}}_o)$, and $\mathbf{G}_M(\mathbf{x}_o)$, respectively, obtained by replacing the unknown constant neural network weights $\{\boldsymbol{\Theta}_\mathbf{D}\}$, $\{\boldsymbol{\Theta}_\mathbf{C}\}$, and $\{\boldsymbol{\Theta}_\mathbf{G}\}$ in (22)–(24) by their estimates $\{\hat{\boldsymbol{\Theta}}_\mathbf{D}\}$, $\{\hat{\boldsymbol{\Theta}}_\mathbf{C}\}$, and $\{\hat{\boldsymbol{\Theta}}_\mathbf{G}\}$, i.e.,

$$\hat{\mathbf{D}}_M(\mathbf{x}_o) = \left[\left\{\hat{\boldsymbol{\Theta}}_\mathbf{D}\right\}^T \bullet \{\boldsymbol{\Xi}_\mathbf{D}(\mathbf{x}_o)\}\right] \tag{29}$$

$$\hat{\mathbf{C}}_M(\mathbf{x}_o, \dot{\mathbf{x}}_o) = \left[\left\{\hat{\boldsymbol{\Theta}}_\mathbf{C}\right\}^T \bullet \{\boldsymbol{\Xi}_\mathbf{C}(\mathbf{x}_o, \dot{\mathbf{x}}_o)\}\right] \tag{30}$$

$$\hat{\mathbf{G}}_M(\mathbf{x}_o) = \left[\left\{\hat{\boldsymbol{\Theta}}_\mathbf{G}\right\}^T \bullet \{\boldsymbol{\Xi}_\mathbf{G}(\mathbf{x}_o)\}\right] \tag{31}$$

Consider the following control law,

$$\boldsymbol{\tau} = \mathbf{J}_e^T(\mathbf{q})(\mathbf{J}_o^T(\mathbf{x}_o))^+ \mathbf{u}_M + \mathbf{J}_e^T(\mathbf{q})\mathbf{F}_{\mathrm{Ic}} \tag{32}$$

where $\mathbf{F}_{\mathrm{Ic}} \in \mathbb{R}^{mn}$ is the internal force control term defined as

$$\mathbf{F}_{\mathrm{Ic}} = \mathbf{F}_{\mathrm{Id}} + \mathbf{K}_I\mathbf{e}_I \qquad \mathbf{K}_I = \mathbf{K}_I^T > 0 \tag{33}$$

$\mathbf{u}_M \in \mathbb{R}^n$ is the motion control law designed as

$$\begin{aligned}\mathbf{u}_M = &\left[\left\{\hat{\boldsymbol{\Theta}}_\mathbf{D}\right\}^T \bullet \{\boldsymbol{\Xi}_\mathbf{D}(\mathbf{x}_o)\}\right]\ddot{\mathbf{x}}_{\mathrm{or}} \\ &+ \left[\left\{\hat{\boldsymbol{\Theta}}_\mathbf{C}\right\}^T \bullet \{\boldsymbol{\Xi}_\mathbf{C}(\mathbf{x}_o, \dot{\mathbf{x}}_o)\}\right]\dot{\mathbf{x}}_{\mathrm{or}} \\ &+ \left[\left\{\hat{\boldsymbol{\Theta}}_\mathbf{G}\right\}^T \bullet \{\boldsymbol{\Xi}_\mathbf{G}(\mathbf{x}_o)\}\right] \\ &+ \mathbf{K}_D\mathbf{r}_o + k_s\,\mathrm{sgn}(\mathbf{r}_o)\end{aligned} \tag{34}$$

with $\mathbf{K}_D = \mathbf{K}_D^T > 0$ and $k_s \geq \|\mathbf{E}(\mathbf{x}_o, \dot{\mathbf{x}}_o, \dot{\mathbf{x}}_{\mathrm{or}}, \ddot{\mathbf{x}}_{\mathrm{or}})\|$ to suppress the inherent neural network reconstruction errors for closed-loop stability, and the asymptotically convergence of the position tracking error. Note that since $\mathbf{F}_{\mathrm{Id}}$ is in the null space of $\mathbf{J}_o^T(\mathbf{x}_o)$, so is $\mathbf{F}_{\mathrm{Ic}}$.

The closed-loop stability properties are summarized in the following theorem.

Theorem 4.1: *Consider the control law* (32) *and* (33) *in closed-loop with multiple manipulators system, if* $\mathbf{K}_D > 0$, $k_s \geq \|\mathbf{E}(\mathbf{x}_o, \dot{\mathbf{x}}_o, \ddot{\mathbf{x}}_{\mathrm{or}}, \ddot{\mathbf{x}}_{\mathrm{or}})\|$ *and the neural network weight vectors are updated by*

$$\dot{\hat{\boldsymbol{\theta}}}_{\mathbf{D}kj} = \boldsymbol{\Gamma}_{\mathbf{D}kj}\boldsymbol{\xi}_{\mathbf{D}kj}(\mathbf{x}_o)\,\ddot{x}_{\mathrm{or}\,j}\,r_{\mathrm{ok}} \tag{35}$$

$$\dot{\hat{\boldsymbol{\theta}}}_{\mathbf{C}kj} = \boldsymbol{\Gamma}_{\mathbf{C}kj}\boldsymbol{\xi}_{Ck}(\mathbf{x}_o, \dot{\mathbf{x}}_o)\,\dot{x}_{\mathrm{or}\,j}\,r_{\mathrm{ok}} \tag{36}$$

$$\dot{\hat{\boldsymbol{\theta}}}_{\mathbf{G}k} = \boldsymbol{\Gamma}_{\mathbf{G}k}\boldsymbol{\xi}_{\mathbf{G}k}(\mathbf{x}_o)\,r_{\mathrm{ok}} \tag{37}$$

where $\boldsymbol{\Gamma}_{\mathbf{D}kj}$, $\boldsymbol{\Gamma}_{\mathbf{C}kj}$, *and* $\boldsymbol{\Gamma}_{Gkj}$ *are constant symmetric positive definite matrices, then, the following holds*

1. the neural network weight estimates $\hat{\boldsymbol{\theta}}_{\mathbf{D}kj}, \hat{\boldsymbol{\theta}}_{\mathbf{C}kj}, \hat{\boldsymbol{\theta}}_{\mathbf{G}k}$ are all bounded, the tracking errors $\mathbf{e}_o$ and $\dot{\mathbf{e}}_o$ asymptotically tend to zero; and
2. the internal force tracking error is bounded and inversely proportional to the norm of the matrix $(I + \mathbf{K}_I)$.

Proof: Substituting (34) into (17), gives the motion error dynamics,

$$\begin{aligned}&\mathbf{D}_M(\mathbf{x}_o)\dot{\mathbf{r}}_o + \mathbf{C}_M(\mathbf{x}_o, \dot{\mathbf{x}}_o)\mathbf{r}_o + \mathbf{K}_D\mathbf{r}_o + k_s\,\mathrm{sgn}(\mathbf{r}_o) \\ &\quad = \left[\left\{\tilde{\boldsymbol{\Theta}}_\mathbf{D}\right\}^T \bullet \{\boldsymbol{\Xi}_\mathbf{D}(\mathbf{x}_o)\}\right]\ddot{\mathbf{x}}_{\mathrm{or}} \\ &\qquad + \left[\left\{\tilde{\boldsymbol{\Theta}}_\mathbf{C}\right\}^T \bullet \{\boldsymbol{\Xi}_\mathbf{C}(\mathbf{x}_o, \dot{\mathbf{x}}_o)\}\right]\dot{\mathbf{x}}_{\mathrm{or}} \\ &\qquad + \left[\left\{\tilde{\boldsymbol{\Theta}}_\mathbf{G}\right\}^T \bullet \{\boldsymbol{\Xi}_\mathbf{G}(\mathbf{x}_o)\}\right] \\ &\qquad + \mathbf{E}(\mathbf{x}_o, \dot{\mathbf{x}}_o, \dot{\mathbf{x}}_{\mathrm{or}}, \ddot{\mathbf{x}}_{\mathrm{or}})\end{aligned} \tag{38}$$

where $\tilde{(\cdot)} = (\cdot) - \hat{(\cdot)}$ is the estimation error.

Consider the nonnegative scalar function,

$$\begin{aligned}V = &\frac{1}{2}\mathbf{r}_o^T\mathbf{D}_M(\mathbf{x}_o)\mathbf{r}_o + \frac{1}{2}\sum_{k=1}^{n}\sum_{j=1}^{n}\tilde{\boldsymbol{\theta}}_{\mathbf{D}kj}^T\boldsymbol{\Gamma}_{\mathbf{D}kj}^{-1}\tilde{\boldsymbol{\theta}}_{\mathbf{D}kj} \\ &+ \frac{1}{2}\sum_{k=1}^{n}\sum_{j=1}^{n}\tilde{\boldsymbol{\theta}}_{\mathbf{C}kj}^T\boldsymbol{\Gamma}_{\mathbf{C}kj}^{-1}\tilde{\boldsymbol{\theta}}_{\mathbf{C}kj} \\ &+ \frac{1}{2}\sum_{k=1}^{n}\tilde{\boldsymbol{\theta}}_{\mathbf{G}k}^T\boldsymbol{\Gamma}_{\mathbf{G}k}^{-1}\tilde{\boldsymbol{\theta}}_{\mathbf{G}k}\end{aligned} \tag{39}$$

Computing the time derivative of V along (38), then applying the skew-symmetric property of $\dot{\mathbf{D}}_M(\mathbf{x}_o) - 2\mathbf{C}_M(\mathbf{x}_o, \dot{\mathbf{x}}_o)$, yields

$$\begin{aligned}\dot{V} = \mathbf{r}_o^T(\mathbf{D}_M(\mathbf{x}_o)\dot{\mathbf{r}}_o + \mathbf{C}_M(\mathbf{x}_o, \dot{\mathbf{x}}_o)\mathbf{r}_o) \\ + \sum_{k=1}^{n}\sum_{j=1}^{n} \tilde{\boldsymbol{\theta}}_{\mathbf{D}kj}^T \boldsymbol{\Gamma}_{\mathbf{D}kj}^{-1} \dot{\tilde{\boldsymbol{\theta}}}_{\mathbf{D}kj} \\ + \sum_{k=1}^{n}\sum_{j=1}^{n} \tilde{\boldsymbol{\theta}}_{\mathbf{C}kj}^T \boldsymbol{\Gamma}_{\mathbf{C}kj}^{-1} \dot{\tilde{\boldsymbol{\theta}}}_{\mathbf{C}kj} \\ + \sum_{k=1}^{n} \tilde{\boldsymbol{\theta}}_{\mathbf{G}k}^T \boldsymbol{\Gamma}_{\mathbf{G}k}^{-1} \dot{\tilde{\boldsymbol{\theta}}}_{\mathbf{G}k}\end{aligned}$$

Substituting (38) into the above equation leads to

$$\begin{aligned}\dot{V} = -\mathbf{r}_o^T \mathbf{K}_D \mathbf{r}_o - k_s \mathbf{r}_o^T \operatorname{sgn}(\mathbf{r}_o) \\ + \mathbf{r}_o^T \left[\left\{ \tilde{\boldsymbol{\Theta}}_{\mathbf{D}} \right\}^T \bullet \{ \boldsymbol{\Xi}_{\mathbf{D}}(\mathbf{x}_o) \} \right] \ddot{\mathbf{x}}_{\mathrm{or}} \\ + \mathbf{r}_o^T \left[\left\{ \tilde{\boldsymbol{\Theta}}_{\mathbf{C}} \right\}^T \bullet \{ \boldsymbol{\Xi}_{\mathbf{C}}(\mathbf{x}_o, \dot{\mathbf{x}}_o) \} \right] \dot{\mathbf{x}}_{\mathrm{or}} \\ + \mathbf{r}_o^T \left[\left\{ \tilde{\boldsymbol{\Theta}}_{\mathbf{G}} \right\}^T \bullet \{ \boldsymbol{\Xi}_{\mathbf{G}}(\mathbf{x}_o) \} \right] \\ + \mathbf{r}_o^T \mathbf{E}\left(\mathbf{x}_o, \dot{\mathbf{x}}_o, \dot{\mathbf{x}}_{\mathrm{or}}, \ddot{\mathbf{x}}_{\mathrm{or}}\right) + \sum_{k=1}^{n}\sum_{j=1}^{n} \tilde{\boldsymbol{\theta}}_{\mathbf{D}kj}^T \boldsymbol{\Gamma}_{\mathbf{D}kj}^{-1} \dot{\tilde{\boldsymbol{\theta}}}_{\mathbf{D}kj} \\ + \sum_{k=1}^{n}\sum_{j=1}^{n} \tilde{\boldsymbol{\theta}}_{\mathbf{C}kj}^T \boldsymbol{\Gamma}_{\mathbf{C}kj}^{-1} \dot{\tilde{\boldsymbol{\theta}}}_{\mathbf{C}kj} + \sum_{k=1}^{n} \tilde{\boldsymbol{\theta}}_{\mathbf{G}k}^T \boldsymbol{\Gamma}_{\mathbf{G}k}^{-1} \dot{\tilde{\boldsymbol{\theta}}}_{\mathbf{G}k} \quad (40)\end{aligned}$$

Noting that

$$\begin{aligned}\mathbf{r}_o^T \left[\left\{ \tilde{\boldsymbol{\Theta}}_{\mathbf{D}} \right\}^T \bullet \{ \boldsymbol{\Xi}_{\mathbf{D}}(\mathbf{x}_o) \} \right] \ddot{\mathbf{x}}_{\mathrm{or}} \\ = \sum_{k=1}^{n}\sum_{j=1}^{n} \tilde{\boldsymbol{\theta}}_{\mathbf{D}kj}^T \boldsymbol{\xi}_{\mathbf{D}kj}(\mathbf{x}_o)\, \ddot{x}_{\mathrm{or}\,j} r_{\mathrm{ok}} \\ \times \mathbf{r}_o^T \left[\left\{ \tilde{\boldsymbol{\Theta}}_{\mathbf{C}} \right\}^T \bullet \{ \boldsymbol{\Xi}_{\mathbf{C}}(\mathbf{x}_o, \dot{\mathbf{x}}_o) \} \right] \dot{\mathbf{x}}_{\mathrm{or}} \\ = \sum_{k=1}^{n}\sum_{j=1}^{n} \tilde{\boldsymbol{\theta}}_{\mathbf{C}kj}^T \boldsymbol{\xi}_{\mathbf{C}kj}(\mathbf{x}_o, \dot{\mathbf{x}}_o)\, \dot{x}_{\mathrm{or}\,j} r_{\mathrm{ok}} \\ \mathbf{r}_o^T \left[\left\{ \tilde{\boldsymbol{\Theta}}_{\mathbf{G}} \right\}^T \bullet \{ \boldsymbol{\Xi}_{\mathbf{G}}(\mathbf{x}_o) \} \right] = \sum_{k=1}^{n} \tilde{\boldsymbol{\theta}}_{\mathbf{G}k}^T \boldsymbol{\xi}_{\mathbf{G}k}(\mathbf{x}_o) r_{\mathrm{ok}}\end{aligned}$$

Eq. (40) becomes

$$\begin{aligned}\dot{V} = -\mathbf{r}_o^T \mathbf{K}_D \mathbf{r}_o - k_s \mathbf{r}_o^T \operatorname{sgn}(\mathbf{r}_o) \\ + \sum_{k=1}^{n}\sum_{j=1}^{n} \tilde{\boldsymbol{\theta}}_{\mathbf{D}kj}^T \boldsymbol{\xi}_{\mathbf{D}kj}(\mathbf{x}_o)\, \ddot{x}_{\mathrm{or}\,j} r_{\mathrm{ok}} \\ + \sum_{k=1}^{n}\sum_{j=1}^{n} \tilde{\boldsymbol{\theta}}_{\mathbf{C}kj}^T \boldsymbol{\xi}_{\mathbf{C}kj}(\mathbf{x}_o, \dot{\mathbf{x}}_o)\, \dot{x}_{\mathrm{or}\,j} r_{\mathrm{ok}} \\ + \sum_{k=1}^{n} \tilde{\boldsymbol{\theta}}_{\mathbf{G}k}^T \boldsymbol{\xi}_{\mathbf{G}k}(\mathbf{x}_o) r_{\mathrm{ok}} + \mathbf{r}_o^T \mathbf{E}\left(\mathbf{x}_o, \dot{\mathbf{x}}_o, \dot{\mathbf{x}}_{\mathrm{or}}, \ddot{\mathbf{x}}_{\mathrm{or}}\right) \\ + \sum_{k=1}^{n}\sum_{j=1}^{n} \tilde{\boldsymbol{\theta}}_{\mathbf{D}kj}^T \boldsymbol{\Gamma}_{\mathbf{D}kj}^{-1} \dot{\tilde{\boldsymbol{\theta}}}_{\mathbf{D}kj} \\ + \sum_{k=1}^{n}\sum_{j=1}^{n} \tilde{\boldsymbol{\theta}}_{\mathbf{C}kj}^T \boldsymbol{\Gamma}_{\mathbf{C}kj}^{-1} \dot{\tilde{\boldsymbol{\theta}}}_{\mathbf{C}kj} + \sum_{k=1}^{n} \tilde{\boldsymbol{\theta}}_{\mathbf{G}k}^T \boldsymbol{\Gamma}_{\mathbf{G}k}^{-1} \dot{\tilde{\boldsymbol{\theta}}}_{\mathbf{G}k} \quad (41)\end{aligned}$$

Substituting the adaptation laws given by (35)–(37) into the above equation, with $k_s \geq \|\mathbf{E}(\mathbf{x}_o, \dot{\mathbf{x}}_o, \dot{\mathbf{x}}_{\mathrm{or}}, \ddot{\mathbf{x}}_{\mathrm{or}})\|$, gives

$$\dot{V} \leq -\mathbf{r}_o^T \mathbf{K}_D \mathbf{r}_o \leq 0 \quad (42)$$

It follows that $0 \leq V(t) \leq V(0)$, $\forall t \geq 0$. Hence $V(t) \in L_\infty$, which implies that $\tilde{\boldsymbol{\theta}}_{\mathbf{D}kj}$, $\tilde{\boldsymbol{\theta}}_{\mathbf{C}kj}$, and $\tilde{\boldsymbol{\theta}}_{\mathbf{G}k}$ are bounded. In other words, $\hat{\boldsymbol{\theta}}_{\mathbf{D}kj}$, $\hat{\boldsymbol{\theta}}_{\mathbf{C}kj}$, and $\hat{\boldsymbol{\theta}}_{\mathbf{G}k}$ are all bounded for $\boldsymbol{\theta}_{\mathbf{D}kj}$, $\boldsymbol{\theta}_{\mathbf{C}kj}$, and $\boldsymbol{\theta}_{\mathbf{G}k}$ are constants though unknown. From (42), it is evident that $\mathbf{r}_o \in L_2^n$. Consequently, from Lemma 4.1, $\mathbf{e}_o \in L_2^n \cap L_\infty^n$, $\mathbf{e}_o$ is continuous and $\mathbf{e}_o \to 0$ as $t \to \infty$, and $\dot{\mathbf{e}}_o \in L_2^n$. By noting that $\mathbf{r}_o \in L_2^n$, $\mathbf{x}_{\mathrm{od}}, \dot{\mathbf{x}}_{\mathrm{od}}, \ddot{\mathbf{x}}_{\mathrm{od}} \in L_\infty^n$, and $\{\boldsymbol{\Xi}_{\mathbf{D}}(\mathbf{x}_o)\}$, $\{\boldsymbol{\Xi}_{\mathbf{C}}(\mathbf{x}_o, \dot{\mathbf{x}}_o)\}$, and $\{\boldsymbol{\Xi}_{\mathbf{G}}(\mathbf{x}_o)\}$ are of bounded basis functions, it is concluded that $\dot{\mathbf{r}}_o \in L_\infty^n$ from Eq. (38). Using the fact that $\mathbf{r}_o \in L_2^n$ and $\dot{\mathbf{r}}_o \in L_\infty^n$, thus $\mathbf{r}_o \to 0$ as $t \to \infty$.[32] Hence $\dot{\mathbf{e}}_o \to 0$ as $t \to \infty$.

The boundedness of the internal force errors can be seen from the following equation derived from (16) and (32) as

$$\mathbf{J}_e^T(\mathbf{q})(\mathbf{F}_{\mathrm{Ic}} - \mathbf{F}_I) = \boldsymbol{\zeta}\left(\mathbf{x}_o, \dot{\mathbf{x}}_o, \dot{\mathbf{x}}_{\mathrm{or}}, \ddot{\mathbf{x}}_{\mathrm{or}}\right) \quad (43)$$

where $\boldsymbol{\zeta}(\mathbf{x}_o, \dot{\mathbf{x}}_o, \dot{\mathbf{x}}_{\mathrm{or}}, \ddot{\mathbf{x}}_{\mathrm{or}})$ is a bounded function defined as

$$\begin{aligned}\boldsymbol{\zeta}\left(\mathbf{x}_o, \dot{\mathbf{x}}_o, \dot{\mathbf{x}}_{\mathrm{or}}, \ddot{\mathbf{x}}_{\mathrm{or}}\right) \\ = \mathbf{D}(\mathbf{q})\mathbf{J}_e^{-1}(\mathbf{q})\mathbf{J}_o(\mathbf{x}_o)\ddot{\mathbf{x}}_o \\ + \mathbf{D}(\mathbf{q})\frac{d}{dt}(\mathbf{J}_e^{-1}(\mathbf{q})\mathbf{J}_o(\mathbf{x}_o))\dot{\mathbf{x}}_o \\ + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\mathbf{J}_e^{-1}(\mathbf{q})\mathbf{J}_o(\mathbf{x}_o)\dot{\mathbf{x}}_o + \mathbf{G}(\mathbf{q}) \\ + \mathbf{J}_e^T(\mathbf{q})(\mathbf{J}_o^T(\mathbf{x}_o))^+ (\mathbf{D}_o(\mathbf{x}_o)\ddot{\mathbf{x}}_o + \mathbf{C}_o(\mathbf{x}_o, \dot{\mathbf{x}}_o)\dot{\mathbf{x}}_o \\ + \mathbf{G}_o(\mathbf{x}_o)) - \mathbf{J}_e^T(\mathbf{q})(\mathbf{J}_o^T(\mathbf{x}_o))^+ \mathbf{u}_M\end{aligned}$$

Substituting for $\mathbf{F}_{\text{Ic}}$ from (33) into (43) and using (26), yields the following internal force tracking error system,

$$\mathbf{e}_I = (I + \mathbf{K}_I)^{-1}\mathbf{J}_e^{-T}(\mathbf{q})\boldsymbol{\zeta}\left(\mathbf{x}_o, \dot{\mathbf{x}}_o, \dot{\mathbf{x}}_{\text{or}}, \ddot{\mathbf{x}}_{\text{or}}\right)$$

It is clear that the force error is bounded and inversely proportional to the gain matrix $(I + \mathbf{K}_I)$. It can be reduced to an arbitrarily small value with sufficiently large gain $\mathbf{K}_I$. ■

With regard to the implementation issues, we have the following remarks to make.

Remark 4.1: From the control law given in (32), it can be seen that $\mathbf{x}_o$ and $\dot{\mathbf{x}}_o$ are needed for its implementation in practice. While it is true that an additional or external sensory system can be used to measure these signals, it is desirable and natural to use existing sensors in the robotic system without an increase in cost. In practice, only one robot, say the ith robot, in the closed chain needs to be chosen for the computation of the required signals using forward kinematics for $\mathbf{x}_o = \mathbf{x}_o(\mathbf{q}_i)$ and Eq. (13) for $\dot{\mathbf{x}}_o = \dot{\mathbf{x}}_o(\mathbf{q}_i, \dot{\mathbf{q}}_i)$. To reduce the computational load further, the inputs to the neural networks can be changed from $\mathbf{x}_o$ and $\dot{\mathbf{x}}_o$ to $\mathbf{q}_i$ and $\dot{\mathbf{q}}_i$ accordingly without affecting the stability properties of the closed-loop system, and no change in the control structure.

Remark 4.2: From (32), it is clear that the controller does not require the inverse of the manipulator Jacobian matrix, $\mathbf{J}_e^{-1}(\mathbf{q})$, which is difficult to obtain especially for a robot with high degrees of freedom.

Remark 4.3: As the complicated nonlinear functions $\mathbf{D}_M(\mathbf{x}_o)$, $\mathbf{C}_M(\mathbf{x}_o, \dot{\mathbf{x}}_o)$, and $\mathbf{G}_M(\mathbf{x}_o)$ given in (18)–(20) are parameterized by neural networks directly, there is no need to go through the tedious process of deriving the regressor matrix, which is necessary in the conventional model-based control approach.

Remark 4.4: The presence of sgn(·) function in the motion control law (34) inevitably introduces chattering, which is undesirable as it may excite mechanical resonance and causes mechanical wear and tear. To alleviate this problem, many approximation mechanism such as boundary layer can be used as suggested in refs. 22 and 33.

5. SIMULATION EXAMPLE

In this section, the proposed adaptive neural network control scheme is applied to coordinated control of two planar three-link manipulators through computer simulation. See Figure 2. The dynamic equation of the ith manipulator is given by (5) as

$$\mathbf{D}_i(\mathbf{q}_i)\ddot{\mathbf{q}}_i + \mathbf{C}_i(\mathbf{q}_i, \dot{\mathbf{q}}_i)\dot{\mathbf{q}}_i + \mathbf{G}_i(\mathbf{q}_i) + \mathbf{J}_{ei}^T(\mathbf{q}_i)\mathbf{F}_{ei} = \boldsymbol{\tau}_i \qquad i = 1, 2$$

where the inertia matrix $\mathbf{D}_i(\mathbf{q}_i)$ is given by

$$\mathbf{D}_i(\mathbf{q}_i) = \begin{bmatrix} d_{i11}(\mathbf{q}_i) & d_{i12}(\mathbf{q}_i) & d_{i13}(\mathbf{q}_i) \\ d_{i21}(\mathbf{q}_i) & d_{i22}(\mathbf{q}_i) & d_{i23}(\mathbf{q}_i) \\ d_{i31}(\mathbf{q}_i) & d_{i32}(\mathbf{q}_i) & d_{i33}(\mathbf{q}_i) \end{bmatrix}$$

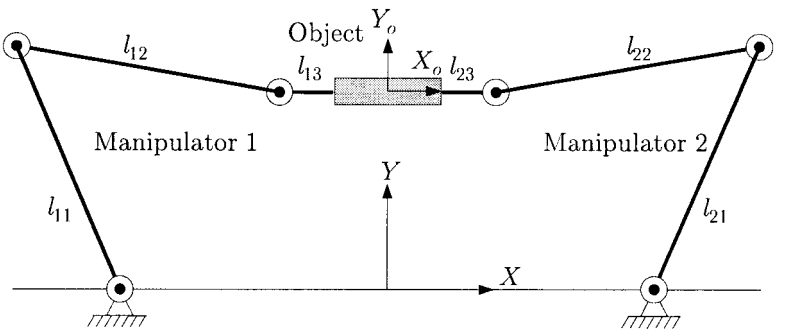

Figure 2. Coordinated control of two planar manipulators.

with

$$
\begin{aligned}
d_{i11}(\mathbf{q}_i) &= p_{i1} + 2\,p_{i2}\cos(q_{i2}) + 2\,p_{i3}\cos(q_{i3}) \\
&\quad + 2\,p_{i4}\cos(q_{i2}+q_{i3}) \\
d_{i12}(\mathbf{q}_i) &= p_{i5} + p_{i2}\cos(q_{i2}) + 2\,p_{i3}\cos(q_{i3}) \\
&\quad + p_{i4}\cos(q_{i2}+q_{i3}) \\
d_{i13}(\mathbf{q}_i) &= p_{i6} + p_{i3}\cos(q_{i3}) + p_{i4}\cos(q_{i2}+q_{i3}) \\
d_{i21}(\mathbf{q}_i) &= d_{i12}(\mathbf{q}_i) \\
d_{i22}(\mathbf{q}_i) &= p_{i5} + 2\,p_{i3}\cos(q_{i3}) \\
d_{i23}(\mathbf{q}_i) &= p_{i6} + p_{i3}\cos(q_{i3}) \\
d_{i31}(\mathbf{q}_i) &= d_{i13}(\mathbf{q}_i) \\
d_{i32}(\mathbf{q}_i) &= d_{i23}(\mathbf{q}_i) \\
d_{i33}(\mathbf{q}_i) &= p_{i6}
\end{aligned}
$$

the Coriolis and centrifugal matrix $\mathbf{C}_i(\mathbf{q}_i, \dot{\mathbf{q}}_i)$ is given by

$$
\mathbf{C}_i(\mathbf{q}_i, \dot{\mathbf{q}}_i) = \begin{bmatrix} c_{i11}(\mathbf{q}_i, \dot{\mathbf{q}}_i) & c_{i12}(\mathbf{q}_i, \dot{\mathbf{q}}_i) & c_{i13}(\mathbf{q}_i, \dot{\mathbf{q}}_i) \\ c_{i21}(\mathbf{q}_i, \dot{\mathbf{q}}_i) & c_{i22}(\mathbf{q}_i, \dot{\mathbf{q}}_i) & c_{i23}(\mathbf{q}_i, \dot{\mathbf{q}}_i) \\ c_{i31}(\mathbf{q}_i, \dot{\mathbf{q}}_i) & c_{i32}(\mathbf{q}_i, \dot{\mathbf{q}}_i) & c_{i33}(\mathbf{q}_i, \dot{\mathbf{q}}_i) \end{bmatrix}
$$

with

$$
\begin{aligned}
c_{i11}(\mathbf{q}_i, \dot{\mathbf{q}}_i) &= -p_{i2}\dot{q}_{i2}\sin(q_{i2}) - p_{i3}\dot{q}_{i3}\sin(q_{i3}) \\
&\quad - p_{i4}(\dot{q}_{i2}+\dot{q}_{i3})\sin(q_{i2}+q_{i3}) \\
c_{i12}(\mathbf{q}_i, \dot{\mathbf{q}}_i) &= -p_{i2}(\dot{q}_{i2}+\dot{q}_{i3})\sin(q_{i2}) - p_{i3}\dot{q}_{i3}\sin(q_{i3}) \\
&\quad - p_{i4}\big(\dot{q}_{i1}+\dot{q}_{i2}+\dot{q}_{i3}\big)\sin(q_{i2}+q_{i3}) \\
c_{i13}(\mathbf{q}_i, \dot{\mathbf{q}}_i) &= -p_{i3}\big(\dot{q}_{i1}+\dot{q}_{i2}+\dot{q}_{i3}\big)\sin(q_{i3}) \\
&\quad - p_{i4}\big(\dot{q}_{i1}+\dot{q}_{i2}+\dot{q}_{i3}\big)\sin(q_{i2}+q_{i3}) \\
c_{i21}(\mathbf{q}_i, \dot{\mathbf{q}}_i) &= p_{i2}\dot{q}_{i1}\sin(q_{i2}) - p_{i3}\dot{q}_{i3}\sin(q_{i3}) \\
&\quad + p_{i4}\dot{q}_{i1}\sin(q_{i2}+q_{i3}) \\
c_{i22}(\mathbf{q}_i, \dot{\mathbf{q}}_i) &= -p_{i3}\dot{q}_{i3}\sin(q_{i3}) \\
c_{i23}(\mathbf{q}_i, \dot{\mathbf{q}}_i) &= -p_{i3}\big(\dot{q}_{i1}+\dot{q}_{i2}+\dot{q}_{i3}\big)\sin(q_{i3}) \\
c_{i31}(\mathbf{q}_i, \dot{\mathbf{q}}_i) &= p_{i3}(\dot{q}_{i1}+\dot{q}_{i2})\sin(q_{i3}) \\
&\quad + p_{i4}\dot{q}_{i1}\sin(q_{i2}+q_{i3}) \\
c_{i32}(\mathbf{q}_i, \dot{\mathbf{q}}_i) &= p_{i3}(\dot{q}_{i1}+\dot{q}_{i2})\sin(q_{i3}) \\
c_{i33}(\mathbf{q}_i, \dot{\mathbf{q}}_i) &= 0
\end{aligned}
$$

the gravitational forces $\mathbf{G}_i(\mathbf{q}_i)$ is given by

$$
\mathbf{G}_i(\mathbf{q}_i) = \begin{bmatrix} g_{i1}(\mathbf{q}_i) \\ g_{i2}(\mathbf{q}_i) \\ g_{i3}(\mathbf{q}_i) \end{bmatrix}
$$

with

$$
\begin{aligned}
g_{i1}(\mathbf{q}_i) &= p_{i7}\cos(q_{i1}) + p_{i8}\cos(q_{i1}+q_{i2}) \\
&\quad + p_{i9}\cos(q_{i1}+q_{i2}+q_{i3}) \\
g_{i2}(\mathbf{q}_i) &= p_{i8}\cos(q_{i1}+q_{i2}) + p_{i9}\cos(q_{i1}+q_{i2}+q_{i3}) \\
g_{i3}(\mathbf{q}_i) &= p_{i9}\cos(q_{i1}+q_{i2}+q_{i3})
\end{aligned}
$$

and

$$
\mathbf{P}_i = [\; p_{i1} \;\; p_{i2} \;\; p_{i3} \;\; p_{i4} \;\; p_{i5} \;\; p_{i6} \;\; p_{i7} \;\; p_{i8} \;\; p_{i9} \;]^T
$$

are the physical parameters of the ith manipulators. The actual values of the parameters used for simulation are

$$
\begin{aligned}
\mathbf{P}_1 = \mathbf{P}_2 = [\; & 4.81 \quad 1.29 \quad 0.05 \quad 0.05 \quad 1.30 \\
& 0.12 \quad 3.62 \quad 1.29 \quad 0.05 \;]^T \ \text{kg m}^2
\end{aligned}
$$

The manipulator Jacobian matrix $\mathbf{J}_{ei}(\mathbf{q}_i)$ is given by

$$
\mathbf{J}_{ei}(\mathbf{q}_i) = \begin{bmatrix} j_{ei11}(\mathbf{q}_i) & j_{ei12}(\mathbf{q}_i) & j_{ei13}(\mathbf{q}_i) \\ j_{ei21}(\mathbf{q}_i) & j_{ei22}(\mathbf{q}_i) & j_{ei23}(\mathbf{q}_i) \\ j_{ei31}(\mathbf{q}_i) & j_{ei32}(\mathbf{q}_i) & j_{ei33}(\mathbf{q}_i) \end{bmatrix}
$$

where

$$
\begin{aligned}
j_{ei11}(\mathbf{q}_i) &= -l_{i1}\sin(q_{i1}) - l_{i2}\sin(q_{i1}+q_{i2}) \\
&\quad - l_{i3}\sin(q_{i1}+q_{i2}+q_{i3}) \\
j_{ei12}(\mathbf{q}_i) &= -l_{i2}\sin(q_{i1}+q_{i2}) - l_{i3}\sin(q_{i1}+q_{i2}+q_{i3}) \\
j_{ei13}(\mathbf{q}_i) &= -l_{i3}\sin(q_{i1}+q_{i2}+q_{i3}) \\
j_{ei21}(\mathbf{q}_i) &= l_{i1}\cos(q_{i1}) + l_{i2}\cos(q_{i1}+q_{i2}) \\
&\quad + l_{i3}\cos(q_{i1}+q_{i2}+q_{i3}) \\
j_{ei22}(\mathbf{q}_i) &= l_{i2}\cos(q_{i1}+q_{i2}) + l_{i3}\cos(q_{i1}+q_{i2}+q_{i3}) \\
j_{ei23}(\mathbf{q}_i) &= l_{i3}\cos(q_{i1}+q_{i2}+q_{i3}) \\
j_{ei31}(\mathbf{q}_i) &= j_{ei32}(\mathbf{q}_i) = j_{ei33}(\mathbf{q}_i) = 1
\end{aligned}
$$

and l_{i1}, l_{i2}, and l_{i3} are the lengths of links 1–3, respectively, of the ith manipulator.

The dynamic equation of the object is given by

$$\mathbf{D}_o(\mathbf{x}_o)\ddot{\mathbf{x}}_o + \mathbf{G}_o(\mathbf{x}_o) = \mathbf{J}_{o1}^T(\mathbf{x}_o)\mathbf{F}_{e1} + \mathbf{J}_{o2}^T(\mathbf{x}_o)\mathbf{F}_{e2}$$

where

$$\mathbf{D}_o(\mathbf{x}_o) = \begin{bmatrix} m_o & 0 & 0 \\ 0 & m_o & 0 \\ 0 & 0 & I_o \end{bmatrix}$$

$$\mathbf{G}_o(\mathbf{x}_o) = \begin{bmatrix} 0 \\ m_o g \\ 0 \end{bmatrix}$$

with m_o and I_o being the mass and inertia of the object, respectively, and

$$\mathbf{J}_{o1}(\mathbf{x}_o) = \begin{bmatrix} 1 & 0 & l_{1l}\sin(x_{o3}) \\ 0 & 1 & -l_{1l}\cos(x_{o3}) \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{J}_{o2}(\mathbf{x}_o) = \begin{bmatrix} 1 & 0 & -l_{2l}\sin(x_{o3}) \\ 0 & 1 & l_{2l}\cos(x_{o3}) \\ 0 & 0 & 1 \end{bmatrix}$$

with l_{il} being the length from the ith end-effector to the center of mass of the object. The actual value of m_o and I_o used for simulation are 5.0 kg and 1.0 kg m^2, respectively.

The desired trajectory for the object is chosen as

$$\mathbf{x}_{\text{od}}(t) = \begin{bmatrix} -0.25\cos(\pi t) \\ 0.75 - 0.25\cos(\pi t) \\ 0.10\sin(\pi t) \end{bmatrix}$$

and the desired internal force are chosen as

$$\mathbf{F}_{\text{Id}} = \begin{bmatrix} 0.0 \\ 0.0 \\ 0.0 \end{bmatrix}$$

Assuming the exact values of the parameter vector $\mathbf{P}$, the mass of the object m_o, and its inertia I_o are unknown, only their estimated values are given, i.e.,

$$\hat{\mathbf{P}}_1 = \hat{\mathbf{P}}_2 = [\,4.70 \quad 1.40 \quad 0.04 \quad 0.04 \quad 1.40 \quad 0.10 \quad 3.40 \quad 1.40 \quad 0.04\,]^T \text{ kg m}^2$$

and

$$\hat{m}_o = 4.0 \text{ kg} \qquad \hat{I}_o = 0.8 \text{ kg m}^2$$

The initial position and velocity of the object are

$$\mathbf{x}_o(0) = [\,-0.30 \quad 0.55 \quad -0.06\,]^T$$

$$\dot{\mathbf{x}}_o(0) = [\,0.0 \quad 0.0 \quad 0.0\,]^T$$

For each element of $\mathbf{D}_M(\mathbf{x}_o)$ and $\mathbf{G}_M(\mathbf{x}_o)$, a neural network with input $\mathbf{x}_o$ is used whereas for each element of $\mathbf{C}_M(\mathbf{x}_o, \dot{\mathbf{x}}_o)$, a neural network with inputs $\mathbf{x}_o$ and $\dot{\mathbf{x}}_o$ is chosen. The center of the localized Gaussian RBFs are evenly distributed to span the input space of the network. The variance of the Gaussian functions are fixed at 10.0. The gains for the controller are chosen as

$$\mathbf{K}_D = \text{diag}[20.0]$$

$$\mathbf{\Lambda} = \text{diag}[10.0]$$

$$\mathbf{K}_I = \text{diag}[0.8]$$

The sliding mode control term is excluded by setting $k_s = 0$ to show the effectiveness of the neural network controller.

5.1. Non-Adaptive Neural Network Control

For the purpose of comparison, consider first the control performance when the weight adaptation laws of the neural network controller given by (35)–(37) are not activated by setting the gains $\mathbf{\Gamma}_{\mathbf{D}kj}$, $\mathbf{\Gamma}_{\mathbf{C}kj}$, and $\mathbf{\Gamma}_{\mathbf{G}k}$ to zero. In this case, the resulting

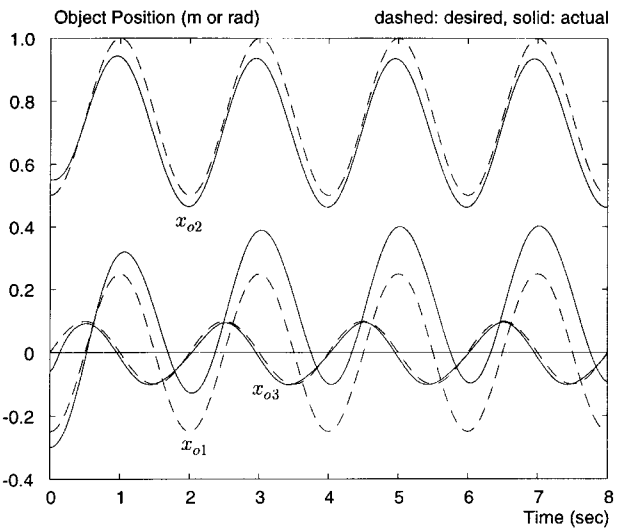

Figure 3. Position tracking of nonadaptive neural network control.

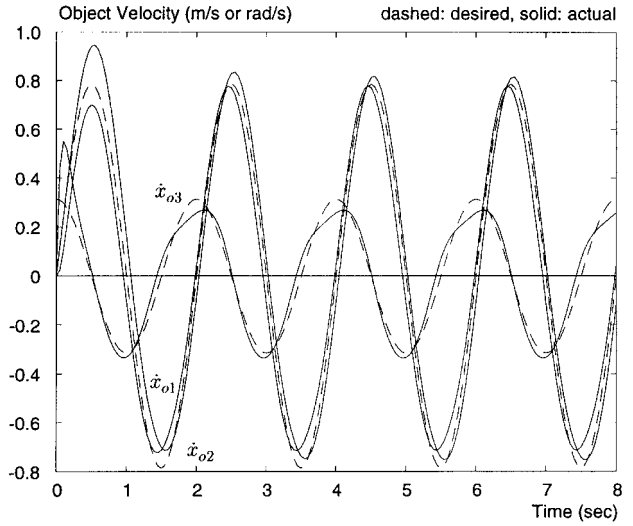

Figure 4. Velocity tracking of nonadaptive neural network control.

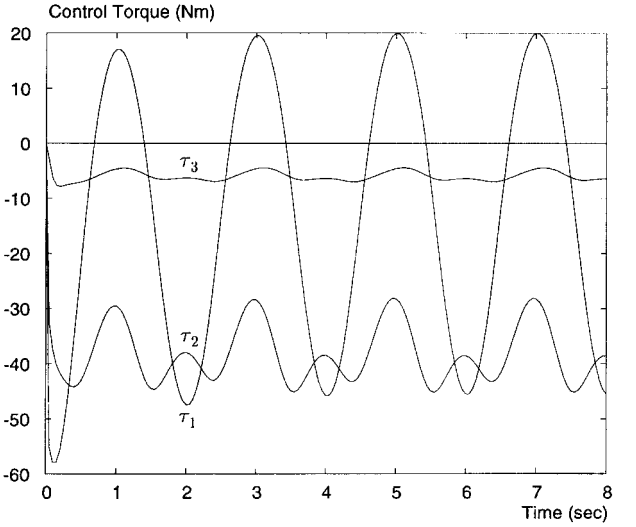

Figure 6. Control signals of nonadaptive neural network control for robot 2.

control action is effectively a fixed model-based control plus a conventional PD control. Figures 3 and 4 show the position and velocity tracking performances of the object. Figures 5 and 6 show the corresponding control signals for the manipulators and the internal force responses are shown in Figures 7 and 8. It can be observed from these results that the nonadaptive neural network control has large tracking errors.

5.2. Adaptive Neural Network Control

In this case, the adaptation gains in (35)–(37) are chosen as follows,

$$\mathbf{\Gamma}_{\mathbf{D}kj} = \text{diag}[0.2]$$

$$\mathbf{\Gamma}_{\mathbf{C}kj} = \text{diag}[0.1]$$

$$\mathbf{\Gamma}_{\mathbf{G}k} = \text{diag}[15.0]$$

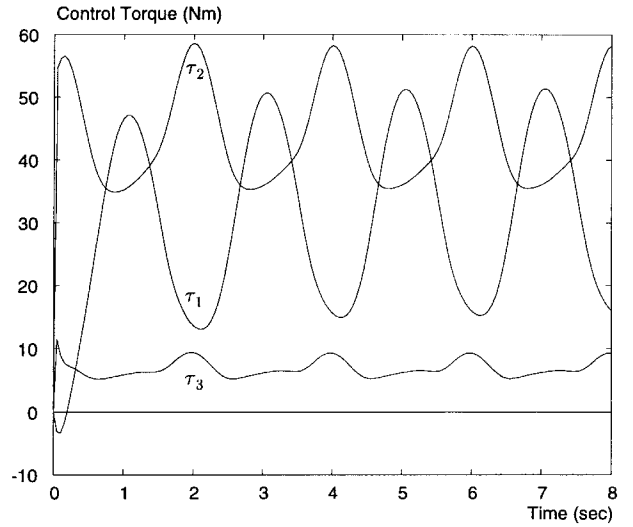

Figure 5. Control signals of nonadaptive neural network control for robot 1.

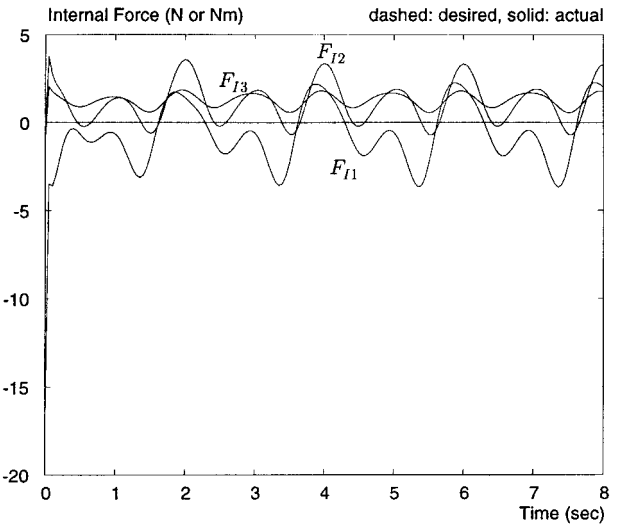

Figure 7. Internal force of nonadaptive neural network control for robot 1.

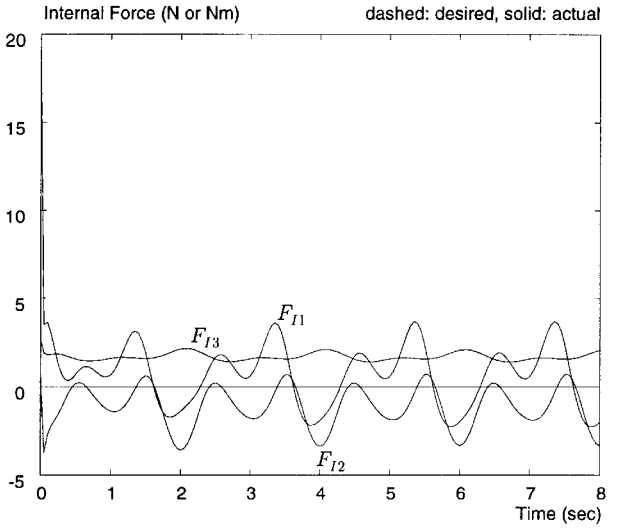

Figure 8. Internal force of nonadaptive neural network control for robot 2.

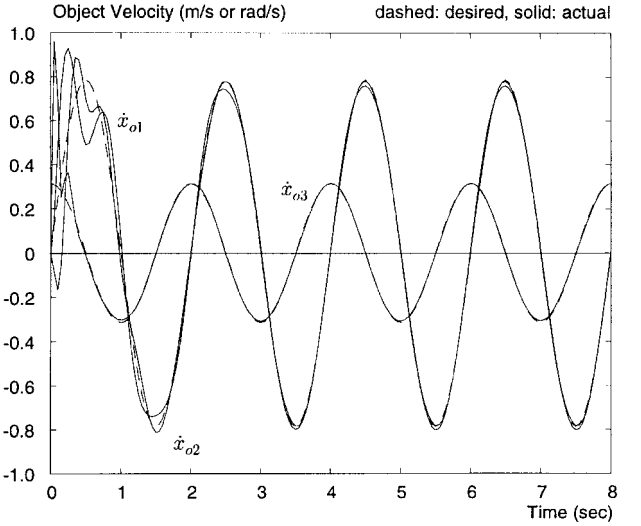

Figure 10. Velocity tracking of adaptive neural network control.

The position and velocity tracking performances of the object are plotted in Figures 9 and 10, respectively, and the control signals are given in Figures 11 and 12. The internal force responses are shown in Figures 13 and 14. It can be seen that the tracking errors are much smaller than the nonadaptive case. The simulation results thus demonstrate that the proposed adaptive neural network control can effectively control the unknown nonlinear dynamic system. By adjusting the adaptation gain and other factors, such as the size of the neural networks, different tracking performance can be achieved. The norms of $\mathbf{D}_M(\mathbf{x}_o)$ and $\hat{\mathbf{D}}_M(\mathbf{x}_o)$, $\mathbf{C}_M(\mathbf{x}_o, \dot{\mathbf{x}}_o)$ and $\hat{\mathbf{C}}_M(\mathbf{x}_o, \dot{\mathbf{x}}_o)$, $\mathbf{G}_M(\mathbf{x}_o)$ and $\hat{\mathbf{G}}_M(\mathbf{x}_o)$ are shown in Figures 15–17. As can be seen, they are all bounded. The simulation results thus demonstrate that the proposed adaptive neural network control can effectively control the unknown nonlinear dynamic system. By adjusting the adaptation gains and the sliding mode gain, different tracking performances

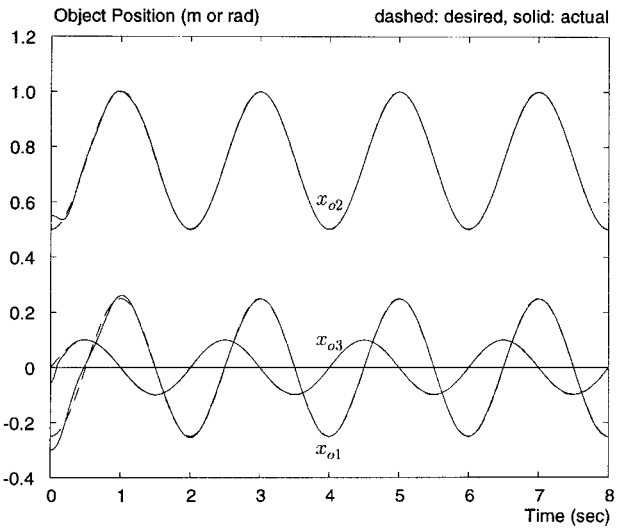

Figure 9. Position tracking of adaptive neural network control.

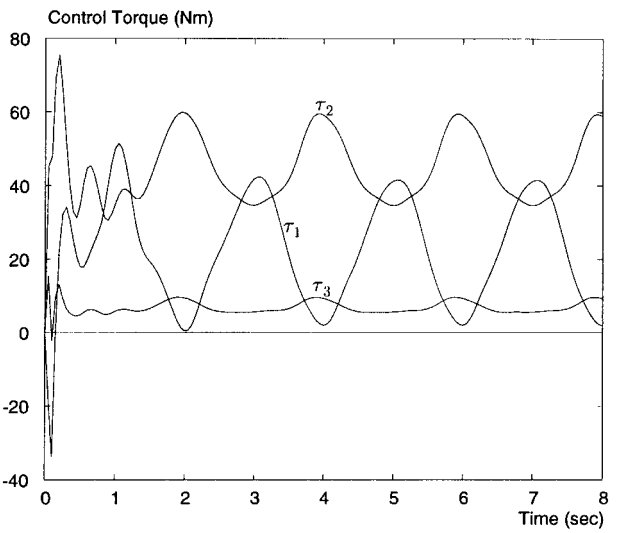

Figure 11. Control signals of adaptive neural network control for robot 1.

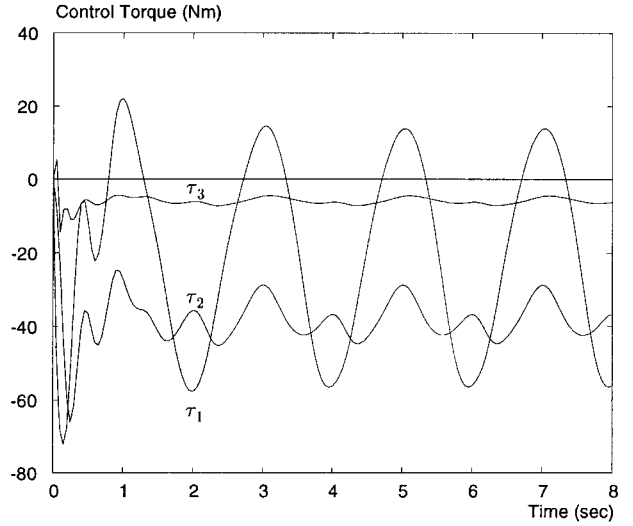

Figure 12. Control signals of adaptive neural network control for robot 2.

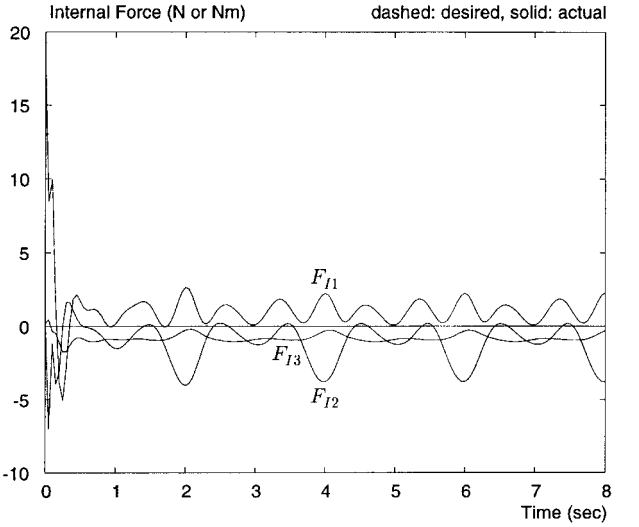

Figure 14. Internal force of adaptive neural network control for robot 2.

can be achieved. Another key factor in determining the control performance is the number of nodes of neural network. Nonetheless, the selection of the size of network is still a current research interest. The numbers of neural network nodes used in this article were chosen by trial and error through computer simulations. It was observed that with a smaller number of neural network nodes, say 50 nodes, the control performance is not that good. When the number of neural network nodes is increased to 200, there is a significant improvement in the control performance. However, when it is increased further, there is no perceptible improvement in the control performance.

6. CONCLUSION

In this article, adaptive neural network control of coordinated multiple manipulators has been considered in an effort to eliminate the time-consuming and error prone dynamic modeling process which is

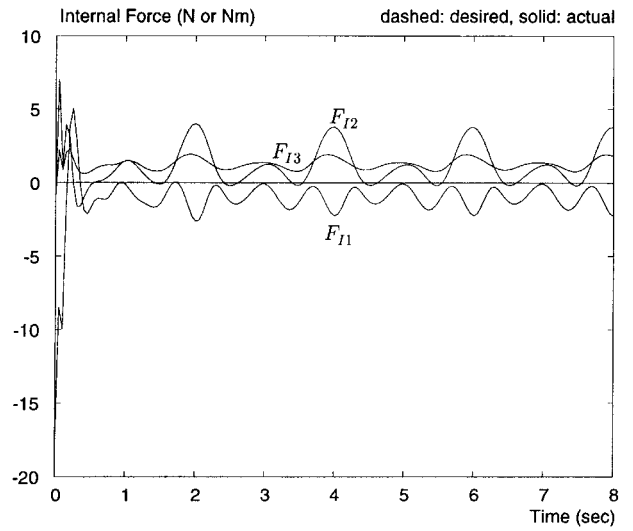

Figure 13. Internal force of adaptive neural network control for robot 1.

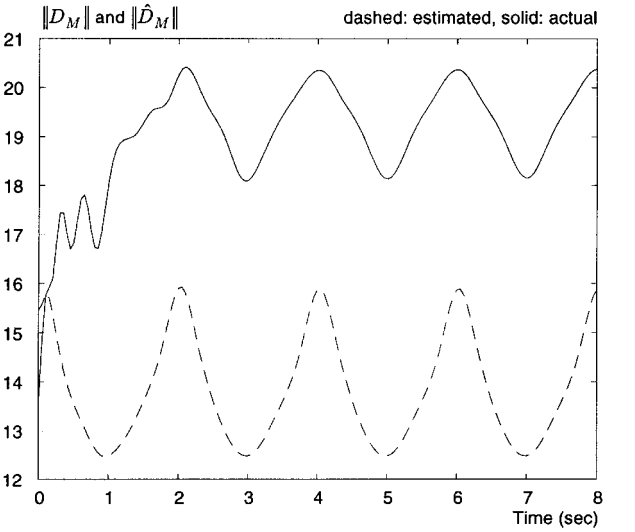

Figure 15. Comparison of $\|\mathbf{D}_M(\mathbf{x}_o)\|$ with $\|\hat{\mathbf{D}}_M(\mathbf{x}_o)\|$ of adaptive neural network control.

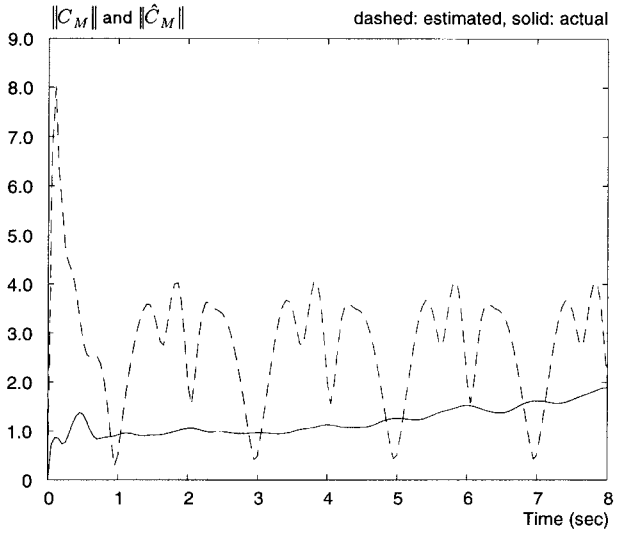

Figure 16. Comparison of $\|\mathbf{C}_M(\mathbf{x}_o, \dot{\mathbf{x}}_o)\|$ with $\|\hat{\mathbf{C}}_M(\mathbf{x}_o, \dot{\mathbf{x}}_o)\|$ of adaptive neural network control.

necessary for the implementation of conventional adaptive control. After a concise dynamic model in the object coordinate space is developed for the coordinated manipulators, an adaptive neural network controller has been presented by combining the techniques of neural network parameterization, adaptive control, and sliding mode control. It has been shown that the motion tracking errors converge to zero asymptotically whereas the internal force tracking error remains bounded and can be made arbitrarily small. Numerical simulations were conducted to show the effectiveness of the proposed method.

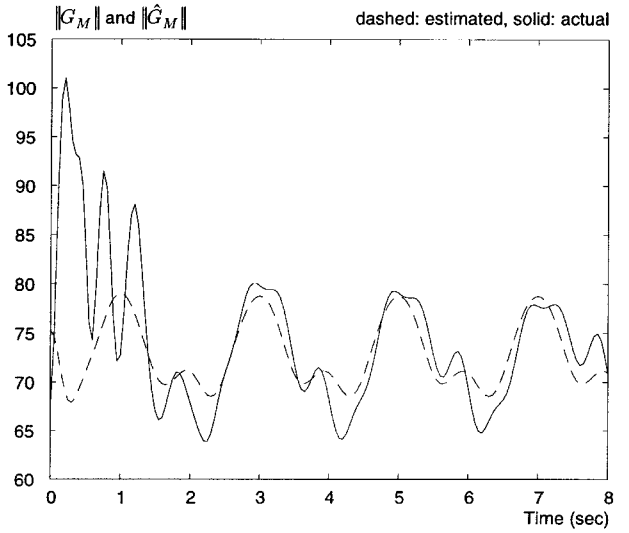

Figure 17. Comparison of $\|\mathbf{G}_M(\mathbf{x}_o)\|$ with $\|\hat{\mathbf{G}}_M(\mathbf{x}_o)\|$ of adaptive neural network control.

REFERENCES

1. M. Zribi and S. Ahmad, Adaptive control for multiple cooperative robot arms, Proc IEEE Conf Decision and Control, Tucson, AZ, 1992, pp. 1392–1398.
2. J.-H. Jean and L.-C. Fu, An adaptive control scheme for coordinated multimanipulator systems, IEEE Trans Robot Automat 9 (1993), 226–231.
3. B. Yao and M. Tomizuka, Adaptive coordinated control of multiple manipulators handling a constrained object, Proc IEEE Int Conf Robotics and Automation, Atlanta, GA, 1993, pp. 624–629.
4. G. Song and L. Cai, A smooth robust control approach to cooperation of multiple robot manipulators, Proc American Control Conf, Seattle, WA, 1995, pp. 1382–1386.
5. C.-Y. Su and Y. Stepanenko, Adaptive sliding mode coordinated control of multiple robot arms attached to a constrained object, IEEE Trans Syst Man Cybern 25 (1995), 871–878.
6. J. Wang, S.J. Dodds, and W.N. Bailey, Co-ordinated control of multiple robotic manipulators handling a common object—theory and experiments, IEE Proc Contr Theory Appl 144 (1997), 73–84.
7. T.J. Tarn, A.K. Bejczy, and X. Yun, Coordinated control of two robot arms, IEEE Int Conf Robotics and Automation, San Francisco, CA, 1986, pp. 1193–1202.
8. J.Y.S. Luh and Y.F. Zheng, Constrained relations between two coordinated industrial robots for motion control, Int J Robot Res 6 (1987), 60–70.
9. S. Arimoto and F. Miyazaki, Cooperative motion control of multiple robot arms or fingers, IEEE Int Conf Robotics and Automation, 1986, pp. 1407–1412.
10. S.A. Hayati, Position and force control of coordinated multiple arms, IEEE Trans Aerosp Electron Syst 24 (1988), 584–590.
11. M. Uchiyama, N. Iwasawa, and K. Hakomri, Hybrid position/force control for coordination of a two-arm robot, IEEE Int Conf Robotics and Automation, Raleigh, NC, 1987, pp. 1242–1247.
12. H. Miyamoto, M. Kawato, T. Setoyama, and R. Suzuki, Feedback error learning neural network for trajectory control of a robotic manipulator, Neural Networks, 1 (1988), 251–265.
13. W.T. Miller, R.P. Hewes, F.H. Galnz, and L.G. Kraft, Real time dynamic control of an industrial manipulator using a neural network based learning controller, IEEE Trans Robot Automat 6 (1990), 1–9.
14. T. Ozaki, T. Suzuki, T. Furuhashi, S. Okuma, and Y. Uchikawa, Trajectory control of robotic manipulators, IEEE Trans Ind Electron 38 (1991), 195–202.
15. M. Saad, P. Bigras, L.A. Dessaint, and K. Al-Haddad, Adaptive robot control using neural networks, IEEE Trans Ind Electron 41 (1994), 173–181.
16. R.M. Sanner and J.-J.E. Slotine, Gaussian networks for direct adaptive control, IEEE Trans Neural Networks 3 (1992), 837–863.

17. E. Tzirkel-Hancock and F. Fallside, Stable control of nonlinear systems using neural networks, Int J Robust Nonlinear Contr 2 (1992), 63–73.
18. F.L. Lewis, K. Liu, and A. Yesildirek, Neural net robot controller with guaranteed tracking performance, IEEE Trans Neural Networks 6 (1995), 703–715.
19. M.M. Polycarpou, Stable adaptive neural network control scheme for nonlinear systems, IEEE Trans Automat Contr 41 (1996), 447–451.
20. S.S. Ge and C.C. Hang, Direct adaptive neural network control of robots, Int J Syst Sci 27 (1996), 533–542.
21. J.T. Spooner and K.M. Passino, Stable adaptive control using fuzzy systems and neural-networks, IEEE Trans Fuzzy Syst 4 (1996), 339–359.
22. S.S. Ge, T.H. Lee, and C.J. Harris, Adaptive neural network control of robot manipulators, World Scientific, London, 1998.
23. T. Poggio and F. Girosi, A theory of networks for approximation and learning, A.I. Memo 1140, Artificial Intelligence Lab, MIT, Cambridge, MA, 1989.
24. F. Girosi and T. Poggio, Networks and the best approximation property, A.I. Memo 1164, Artificial Intelligence Lab, MIT, Cambridge, MA, 1989.
25. T. Poggio and G. Girosi, Networks for approximation and learning, Proc IEEE 78 (1990), 1481–1497.
26. S.S. Ge, C.C. Hang, and L.C. Woon, Adaptive neural network control of robot manipulators in task space, IEEE Trans Ind Electron 44 (1997), 746–752.
27. S.S. Agaian, Hadamard matrices and their applications, Springer-Verlag, New York, 1985.
28. A.A. Cole, J.E. Hauser, and S.S. Sastry, Kinematics and control of multifingered hands with rolling contact, IEEE Trans Automat Contr 34 (1989), 398–404.
29. A.A. Cole, P. Hsu, and S.S. Sastry, Dynamic control of sliding by robot hands for regrasping, IEEE Trans Robot Automat 8 (1992), 42–52.
30. M.W. Spong and M. Vidyasagar, Robot dynamics and control, Wiley, New York, 1989.
31. F.L. Lewis, C.T. Abdallah, and D.M. Dawson, Control of robot manipulators, MacMillan, New York, 1993.
32. C.A. Desoer and M. Vidyasagar, Feedback systems: Input–output properties, Academic Press, New York, 1975.
33. J.-J.E. Slotine and W. Li, On the adaptive control of robot manipulators, Int J Robot Res 6 (1987), 49–59.