Constrained Motion Control of Flexible Robot Manipulators Based on Recurrent Neural Networks

Lianfang Tian, Jun Wang, *Senior Member, IEEE*, and Zongyuan Mao

***Abstract*—In this paper, a neural network approach is presented for the motion control of constrained flexible manipulators, where both the contact force exerted by the flexible manipulator and the position of the end-effector contacting with a surface are controlled. The dynamic equations for vibration of flexible link and constrained force are derived. The developed control scheme can adaptively estimate the underlying dynamics of the manipulator using recurrent neural networks (RNNs). Based on the error dynamics of a feedback controller, a learning rule for updating the connection weights of the adaptive RNN model is obtained. Local stability properties of the control system are discussed. Simulation results are elaborated on for both position and force trajectory tracking tasks in the presence of varying parameters and unknown dynamics, which show that the designed controller performs remarkably well.**

***Index Terms*—Constrained motion, flexible robot manipulators, hybrid position/force control, recurrent neural network.**

I. Introduction

With the advancement of robot technologies and the development of robotic applications, industrial robots can realize many kinds of manipulation tasks such as, assembly, deburring, grinding, to name a few. Many of these applications require contacts to be made with the environment by the end-effector of the robot manipulator. For such tasks, if only position control is considered, the accuracy requirement of the robot control may not be met. Therefore, force control should also be required in addition to position control; namely, both the position control and the force control must be carried out simultaneously [1], [2]. This research topic has received much attention in the robot control literature in recent years.

In addition, the demands for lightweight structure, energy efficiency, and high speed motion have increased recently for robot manipulators to carry out complex industrial tasks. For such occasions, the flexibility of the mechanical structures of robot manipulators is very important for the design of their control systems [3]–[5]. Flexible manipulators exhibit many advantages over rigid robots, i.e., they require less material, are lighter in weight, consume less power, require smaller actuators, are more maneuverable and transportable, have less overall cost, and high payload to robot weight ratio [6]–[8]. However, unlike rigid robot arms, control interaction in flexible robot arms creates a severe stability problem, because of their high nonlinearity and high dimensionality.

The design of control system for robots has played an important role in maintaining high control accuracy. However, there are still many problems to be solved before flexible robots are widely used in industrial tasks, such as severe nonlinearity, coupling due to flexibility of manipulators, uncertainties of constrained environment, changes of system parameters and external payloads, friction in robot joints, gravity, disturbance, and so on. Any of these can lead to instability of a robot system if the robot controller is not designed properly. In other words, those problems will bring difficulties for the design of robot control systems.

The study on the controller design for constrained flexible robots, therefore, has profound importance for practical applications in industries. The conventional approaches to the design of an automatic control system often involve the construction of a mathematical model describing the dynamic behavior of the plant to be controlled and the application of analytical techniques to this plant model to derive an appropriate control law. Usually, such a mathematical model consists of a set of linear or nonlinear differential/difference equations, most of which are derived using some forms of approximation and simplification. The traditional model-based control techniques may fail, however, when a representative model is difficult to obtain, or not accurate, due to uncertainty or sheer complexity. This pushes the control system designers to seek alternatives to the conventional control methods and to look for solutions elsewhere.

With the resurgence of research on intelligent control for various problems of nonlinear nature in recent years, many researchers have applied neurologically inspired approaches for robot control [9]–[13]. The massive parallelism, natural fault-tolerance, and implicit programming of neural network computing architectures, both nonlinear and mapping, suggest they may be good candidates for implementing real-time adaptive control for large scale nonlinear dynamic systems, especially for sophisticated robotic systems [15]–[18]. The major advantage of neural networks, compared with traditional control methods, is that they require no *a priori* knowledge about the controlled system. Rather, they involve using neural networks to complete modeling and control task autonomously. Researchers from different disciplines have published extensively on the use of the neuromorphic models for the control of the nonlinear dynamic robotic systems in the past years, and many promising results in this field are realized [19]–[22].

Manuscript received March 7, 2003; revised October 22, 2003. This work was supported by the Chinese National 863 Plan under Grant 9805-19 and the Hong Kong Research Grants Council under Grant CUHK4174/00E.

J. Wang is with the Department of Automation and Computer-Aided Engineering, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong (e-mail: jwang@acae.cuhk.edu.hk).

L. Tian and Z. Mao are with the Department of Automatic Control Engineering, South China University of Technology, Guangzhou, China (email: auzymao@scut.edu.cn).

Digital Object Identifier 10.1109/TSMCB.2004.826400

Song *et al.* [23] designed a nonlinear predictive controller on the basis of a neural network plant model using the receding-horizon control approach, the control is calculated by minimizing a projected cost function that penalizes future tracking errors. Lin *et al.* [24] gave an approach combining the *a priori* knowledge of the corresponding rigid manipulator's model with two multilayered neural networks for the identification and control of a flexible-link manipulator. The suggested method can use fewer neurons and needs shorter learning time for reducing the error. Kiguchi [25] proposed a fuzzy vector method that enables the controller to deal with force sensor signals which include noise and/or unknown vibrations caused by working tool. Based on the fuzzy vector concept, a fuzzy neural position/force controller is developed. It is utilized to apply force to the unknown environment as it adjusts the force control direction using the fuzzy vector method, and to follow the geometrically unknown surface of the object. A stable discrete-time tracking control approach based on dynamics inversion using dynamic neural networks (DNNs) was developed by Sun. *et al.* [26]. The robot control law is composed of the dynamic inversion of the DNN, adaptive compensation and the DNN variable structure control (VSC) components. The control scheme can guarantee the global stability and tracking error convergence of the DNN control system. In [27], a feedforward neural network is used to adaptively compensate for the uncertainties of the robot dynamics. The connection weights of the neural network are tuned online with no offline learning phase required. It is able to deal with both the modeled and unmodeled uncertainties of the robotic systems. Jung, *et al.* [28], [29] used a neural network (NN) controller to compensate for the ill effects of model uncertainties. Effects on system performance for different choices of the NN input types, hidden neurons, weight update rates, and initial weight values are also investigated extensively.

In this study, through deep and comprehensive analysis, an effective controller based on recurrent neural networks (RNNs) for constrained flexible manipulators is designed. It can maintain the stability of the robot system during the overall process of execution, and also can improve the performance of control system and enhance the adaptive capability against the uncertainties from both internal parameters and external environment.

The remaining of the paper is arranged as follows. Section II gives the dynamic equations of a two-DOF constrained flexible manipulator and its reduced-order model. Section III describes the RNN model and its learning rule. Section IV discusses the results of stability and convergence analysis. Section V elaborate on the force tracking and composite control law. Simulation results under various conditions and environments are discussed in Section VI. Finally, conclusions are drawn in Section VII.

II. Constrained Dynamic Models

Consider a two-DOF manipulator driven by two motors in a horizontal plane. The first link is assumed to be rigid while the second is flexible. The first (rigid) link, having length L_1 and moment of inertia J_1, is clamped on a vertical shaft of the first motor (motor 1) at one end. The second motor drives the second (flexible) link at the other end, as shown in Fig. 1. Let J_h and M_h be the moment of inertia and mass of the rotor of motor 2, respectively. The flexible link, having length L_2 and uniform mass density ρ per unit length, and uniform flexural rigidity EI, is clamped on the vertical shaft of the second motor at one end and has a concentrated mass M_2 at the other end. Let (X, Y) designate an inertial Cartesian coordinate variables and let the constraint surface be described as

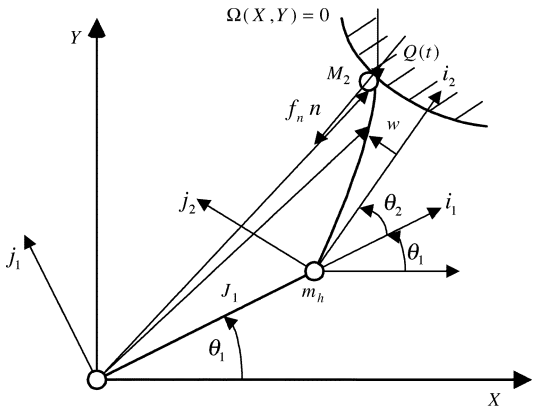

Fig. 1. Configuration of a two-DOF flexible manipulator.

$$\Omega(X, Y) = 0. \tag{1}$$

The end point position vector (X_p, Y_p) of the flexible manipulator can be expressed by rotation angles (θ_1, θ_2) as follows:

$$X_p = L_1 \cos\theta_1 + L_2 \cos(\theta_1 + \theta_2) - \omega_E \cos(\theta_1 + \theta_2) \tag{2}$$

$$Y_p = L_1 \sin\theta_1 + L_2 \sin(\theta_1 + \theta_2) + \omega_E \sin(\theta_1 + \theta_2) \tag{3}$$

where ω_E denotes the transverse displacement at the end point.

Substituting (2) into (1), we can obtain the constraint equation as follows:

$$\Omega(\theta_1, \theta_2, \omega_E) = \Omega(\theta) = 0. \tag{4}$$

The flexible deflection of the second link can be described as

$$\omega(r, t) = \sum_{i=1}^{n} \psi_i(r) q_i(t) \tag{5}$$

where n denotes the number of significant modes, $\psi_i(r)$ $(i = 1, 2, \ldots, n)$ denote the mode shape functions, $q_i(t)$ express time dependent generalized coordinates. In the literature, different choices of spatial basis functions, including the mode shapes of fixed-free [30], pinned-free [31] and a combination of fixed-free and fixed-hinged beam [32], have been made. However, research related to an optical choice of basis functions for reduced-order modeling of flexural robotic systems of general configuration remains to be done. Of course, a good choice of basis functions can lead to an accurate model with smaller number of flexural modes which simplifies the controller design problem. We consider the low-order accurate dynamic modeling as a separate problem. In this study, we chose the mode shapes ψ_i of a clamped-free as an illustration.

Based on the previous discussions, using the Lagrange approach, the dynamic motion equation of the manipulator is given as follows.

$$\begin{bmatrix} M_{11}(z) & M_{12}(z) \\ M_{21}(z) & M_{22}(z) \end{bmatrix} \begin{bmatrix} \ddot{\theta} \\ \ddot{q} \end{bmatrix} + \begin{bmatrix} F_1(z,\dot{z}) \\ F_2(z,\dot{z}) \end{bmatrix} + \begin{bmatrix} 0 \\ K(z) \end{bmatrix} \begin{bmatrix} \theta \\ q \end{bmatrix} = \begin{bmatrix} \tau \\ 0 \end{bmatrix} + \lambda \begin{bmatrix} \frac{\partial \Omega}{\partial \theta} \\ 0 \end{bmatrix} \tag{6}$$

where $z = [\theta^T, q^T]^T$ is the vector of generalized coordinates, $M_{ij}(z)(i,j = 1,2)$ is the positive definite symmetric inertial matrix, $F(z,\dot{z}) = [F_1^T, F_2^T]^T$ consists of the Coriolis and centrifugal forces, τ is the vector of the generalized forces applied to each joint, λ is the Lagrange multiplier associated with the constraint. $K(z)$ is an $n \times n$ elasticity matrix. Noting that the potential energy only includes the strain energy and the gravitational effect in this derivation, the longitudinal and torsional deformations are neglected.

Equation (6) can be transformed into two parts in the following form:

$$M_{11}(z)\ddot{\theta} + M_{12}(z)\ddot{q} + F_1(z,\dot{z}) = \tau + \lambda \frac{\partial \Omega(\theta)}{\partial \theta} \tag{7}$$

$$M_{21}(z)\ddot{\theta} + M_{22}(z)\ddot{q} + F_2(z,\dot{z}) + K(z)q = 0. \tag{8}$$

From (8), we can get

$$\ddot{q} = -M_{22}^{-1}(z)\left[M_{21}(z)\ddot{\theta} + F_2(z,\dot{z}) + K(z)q\right]. \tag{9}$$

Substituting (9) into (7), we have

$$\bar{M}\ddot{\theta} + \bar{C} = \tau + \lambda \frac{\partial \Omega(\theta)}{\partial \theta} \tag{10}$$

where $\bar{M} = M_{11} - M_{12}M_{22}^{-1}M_{21}$, $\bar{C} = F_1 - M_{12}M_{22}^{-1}(F_2 + Kq)$

Because the end-effector of the flexible manipulator is restricted to move on the constraint in the whole process, θ_1 and θ_2 can not be independent in a nonredundant manipulator, therefore, one of them can be expressed by the other one. Without loss of generality, θ_2 is described as the function of θ_1. So we have

$$\theta_2 = \Psi(\theta_1) \tag{11}$$

where ω_E is omitted, as it can be calculated by employing (5) and (8).

To facilitate our subsequent analysis, we reformulate (10) by introducing a new set of variables $\nu = [\nu_1, \nu_2]^T$, as defined in [33] and [34]

$$\nu = \begin{bmatrix} \nu_1 \\ \nu_2 \end{bmatrix} = \begin{bmatrix} \theta_1 \\ \theta_2 - \Psi(\theta_1) \end{bmatrix} = T(\theta). \tag{12}$$

So

$$\theta = \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} = T^{-1}(\nu) = \begin{bmatrix} \nu_1 \\ \nu_2 + \Psi(\nu_1) \end{bmatrix}. \tag{13}$$

Thus, the transformation matrix is given by

$$T_r = \begin{bmatrix} 1 & 0 \\ \frac{\partial \Psi(\nu_1)}{\partial \nu_1} & 1 \end{bmatrix}. \tag{14}$$

After the substitution of (13) into (10), and multiplication of T_r from left, the reduced-order model can be obtained as

$$M^*(\nu)\ddot{\nu} + C^*(\nu, q, \dot{\nu}, \dot{q}) = T_r^T\tau + T_r^T\xi\lambda \tag{15}$$

where $M^* = T_r^T\bar{M}T_r$, $C^* = T_r^T(\bar{M}\dot{T}_r\dot{\nu} + \bar{C})$, $\xi = \partial\Psi(\nu_1)/\partial\nu_1$, and note that $\ddot{\nu}_2 = \dot{\nu}_2 = \nu_2 = 0$.

Through a series of complicated transformations, the motion equation can be decomposed into the following form

$$\ddot{\nu}_1 = (m_{11}^* - pm_{21}^*)^{-1}[\tau_1 + \xi\tau_2 - p\tau_2 + (pc_2^* - c_1^*)] \tag{16}$$

$$\lambda = b_2^{-1}[m_{21}^*\ddot{\nu}_1 + c_2^* - \tau_2] \tag{17}$$

where $m_{ij}(i = 1,2; j = 1,2)$ are the elements of matrix M^*, $b_1 = \partial\Omega(\theta)/\partial\theta_1$, $b_2 = \partial\Omega(\theta)/\partial\theta_2$, $p = (b_1 + \xi b_2)b_2^{-1}$.

The composite controller of the two-DOF flexible manipulator can be designed according to (16) and (17).

III. RNN-Based Robot Control

A. Adaptive Control Scheme

The incredible learning and adaptive capability of biological neuronal mechanisms have inspired many scientists and engineers to apply control methodologies on the biological counterparts. In the efforts to understand the biological control aspects, artificial neural networks have offered the most exciting avenues for new technologies in the intelligent control paradigms [35], [36]. It is well known that a multilayer neural network model is basically a nonlinear extension of a linear adaptive model. It possesses so many advantages in the controller design, compared with conventional control methods, such as the capability of approximating arbitrary nonlinear functions, fault-tolerance, parallel computing and so on. Therefore, the applications of neural networks have received considerable attentions in the control of complex robotic systems [37]–[41].

As an extension of static neural networks, recurrent neural networks (RNNs), which contain a state feedback, may provide more computational advantages than a feedforward neural networks [42], [43]. A nonlinear RNN structure is particularly appropriate for identification, control and filtering applications due to its ability of distributed and multiple super-imposed information processing as a biologically plausible neural system. In these neural machines, the physics of the machine and algorithm of computation are intimately related. Next, we will discuss the proposed controller architecture of flexible manipulators.

Formally speaking, (16) represents the direct dynamics of the controlled manipulators in the reduced-order variable form; namely, it actually refers to a nonlinear transformation (mapping) from inputs (joint torques τ of the manipulator to the outputs (joint motion). In brevity, (16) can be written as follows:

$$\ddot{\nu} = V(\nu, \dot{\nu}, \tau). \tag{18}$$

Then, based on the argument and (10), the inverse dynamics of the manipulator can be obtained as

$$\tau = V^{-1}(\nu, \dot{\nu}, \ddot{\nu}) \tag{19}$$

where $V^{-1}(*)$ denotes the inverse transformation obtained by inverting the robot direct dynamics $V(*)$. These nonlinear transformations are time-dependent. However, in the ensuing discussions, the arguments of $V(*)$ and $V^{-1}(*)$ and their variants will sometimes be dropped for the brevity of notation.

If an accurate dynamic formulation of the manipulator and its accurate parameters are available, then a controller can directly be designed by employing the conventional control methods (such as PID, adaptive control, robust control), all of which are based on computed torque scheme. However, if such a model is not available, the system dynamics has to be adaptively identified in order to achieve a feedforward compensation. In this study, the robot manipulators are a highly nonlinear and heavily coupled complex systems. Therefore, its accurate dynamic model is difficult to obtain. In such case, an RNN model is considered to effectively approximate the inverse mapping $V^{-1}(*)$ as closely as possible [44], [45]. The inverse dynamics of a manipulator, described by the nonlinear transformation $V^{-1}(*)$, can be decomposed into n transformations given in the following form:

$$\tau = V^{-1}(\nu, \dot{\nu}, \ddot{\nu}) = \begin{bmatrix} g_1^{-1}(\nu, \dot{\nu}, \ddot{\nu}) \\ \vdots \\ g_n^{-1}(\nu, \dot{\nu}, \ddot{\nu}) \end{bmatrix} \tag{20}$$

where each $g_i^{-1}(i = 1, \cdots, n)$ defines the inverse dynamics of the corresponding joint, n is the number of joints of the manipulator.

According to the analysis in [41] and [44], here, we can employ a diagonal RNN, which has a feedback in its hidden layer, to model each entry $g_i^{-1}(*)$ of the vector function $V^{-1}(*)$. Therefore, the inverse dynamics model of the overall system can be represented by

$$\tau = [\bar{V}]^{-1}(\nu, \dot{\nu}, \ddot{\nu}) = \begin{bmatrix} N_1^{-1}(\nu, \dot{\nu}, \ddot{\nu}, \omega) \\ \vdots \\ g_n^{-1}(\nu, \dot{\nu}, \ddot{\nu}, \omega) \end{bmatrix} \tag{21}$$

where $[\bar{*}]$ denotes the estimated model, and $N_i(*)(i = 1, \cdots, n)$ represents the output of the each RNN model that is employed to model the inverse dynamics of the manipulators, ω is denoted as the vector of the adjustable weights of the corresponding RNN model and will be defined explicitly in the sequel.

Note that, using a RNN to model the dynamics of a system can basically realize a direct implicit transformation from the joint position vector to the joint torques. This model does not convey any explicit information on the estimated dynamic components of the manipulator such as the inertia matrix. Hence, a direct adaptive control architecture which would be based on a computed torque-like model is difficult to obtain, a neural network model is utilized to approximate the dynamics of the manipulator. Then, the feedforward torques can be used to combine with a feedback control signal to obtain the torques that will finally drive the motors. Therefore, the control scheme can be given as

$$\tau = [\bar{V}]^{-1}(*) + K_p e + K_d \dot{e} = N(*) + K_p e + K_d \dot{e} \tag{22}$$

where $[\bar{V}]^{-1}(*)$ is the neural network approximation of the actual inverse dynamics of the manipulators, $K_p \in R^{n \times m}$ and $K_d \in R^{n \times m}$ are the diagonal gain matrices with entries K_p and K_d, respectively, denoting a servo feedback that is introduced to stabilize the system. $e = \theta_d - \theta_a$, $\dot{e} = \dot{\theta}_d - \dot{\theta}_a$, representing the tracking error and velocity error, respectively. $\theta_d \in R^2$ is defined as the desired trajectories, $\theta_a \in R^2$ as the actual outputs,

Now, making use of (20) and (21), it can be given as

$$N(\nu, \dot{\nu}, \ddot{\nu}, \omega) + K_p e + K_d \dot{e} = V^{-1}(\nu, \dot{\nu}, \ddot{\nu}) \tag{23}$$

$$K_p e + K_d \dot{e} = V^{-1}(\nu, \dot{\nu}, \ddot{\nu}) - N(\nu, \dot{\nu}, \ddot{\nu}, \omega) = \bar{N}(\nu, \dot{\nu}, \ddot{\nu}, \omega) \tag{24}$$

Equation (24) characterizes a decoupled linear system driven by the nonlinear vector function $\bar{N}(\nu, \dot{\nu}, \ddot{\nu}, \omega) \in R^n$. This function represents the error between the actual inverse dynamics $V^{-1}(*)$ and its estimated model $N(*)$ and can be explicitly written as

$$\begin{aligned} \bar{N}(\nu, \dot{\nu}, \ddot{\nu}, \omega) &= \begin{bmatrix} \bar{N}_1(\nu, \dot{\nu}, \ddot{\nu}, \omega) \\ \vdots \\ \bar{N}_n(\nu, \dot{\nu}, \ddot{\nu}, \omega) \end{bmatrix} \\ &= \begin{bmatrix} g_1^{-1}(\nu, \dot{\nu}, \ddot{\nu}) - N_1(\nu, \dot{\nu}, \ddot{\nu}, \omega) \\ \vdots \\ g_n^{-1}(\nu, \dot{\nu}, \ddot{\nu}) - N_n(\nu, \dot{\nu}, \ddot{\nu}, \omega) \end{bmatrix} \end{aligned} \tag{25}$$

where $N_i(\nu, \dot{\nu}, \ddot{\nu}, \omega)$, with $i = 1, 2, \cdots, n$ denotes the error in inverse dynamic modeling for each joint.

It makes intuitive sense that instead of using one neural network to approximate the inverse dynamics of the whole manipulator system, one ought to use a separate network for each joint of the manipulator. With this in mind and by using (25), the error equation for the ith joint of the manipulators is expressed as

$$K_{ip} e_i + K_{id} \dot{e}_i = \bar{N}_i(\nu, \dot{\nu}, \ddot{\nu}, \omega) \quad i = 1, 2, \cdots, n \tag{26}$$

where e_i and $\dot{e}_i$ denote the position and velocity errors at the i-th joint, respectively, k_{ip}, k_{id} are the constant servo gains accordingly, $\bar{N}(\nu, \dot{\nu}, \ddot{\nu}, \omega)$ is the local approximation error of the RNN assigned to the i-th joint. Hence, let us define a time-varying surface $R(t)$ in the e_i and $\dot{e}_i$ space, it gives as follows:

$$R(t) : r(e, \dot{e}, t) = 0 \tag{27}$$

with $r(t) = r(e, \dot{e}) = k_{ip} e_i + k_{id} \dot{e}_i$. Therefore, the scalar signal $r(t)$ can be considered as the distance to the surface $R(t)$. The problem of tracking is then equivalent to that of minimizing the distance to the surface defined by $r(t) = 0$. Now, let us define $\varepsilon_i = k_{ip} e + k_{id} \dot{e}_i$, then (26) can be written as

$$\varepsilon_i = \bar{N}_i(\nu, \dot{\nu}, \ddot{\nu}, \omega). \tag{28}$$

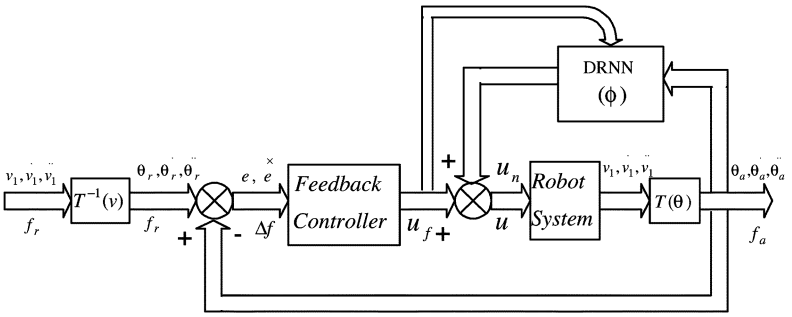

Fig. 2. Diagram of the controller architecture.

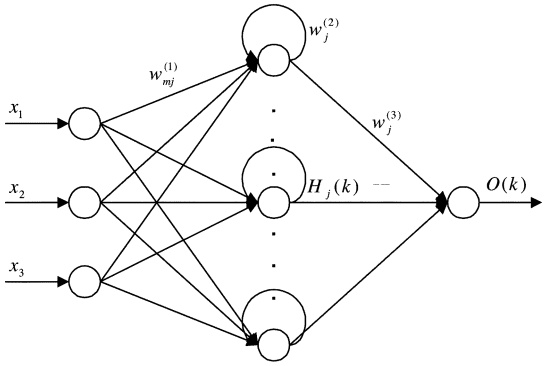

Fig. 3. Architecture of a diagonal RNN model.

Noting that $\varepsilon_i = \varepsilon_i(t)$, one can conclude that $\varepsilon_i(t)$ constitutes a measure of the tracking error that reflects the mismatch between the actual inverse dynamics of the manipulators and its local neural network approximation. The block diagram of the RNN-based robot control system is shown in Fig. 2.

The question that remains to be answered at this point is how to update, or modify, the weights of the RNN model online, so that the error measure $\varepsilon_i(t)$ converges to 0 as $t \to \infty$. Here, a gradient-descent approach comes immediately to mind. To do this, let us define the following cost function $E_i(t)$ for each joint of the manipulators at each sampling instant [41]

$$E_i(t) = \frac{1}{2}\varepsilon_i^2. \tag{29}$$

This cost function gives the squared distance to the surface $r(t) = 0$. Minimizing the cost function $E_i(t)$ over the weight space of the corresponding RNN model forms the basis of the weights updating algorithm. In doing so, let us derive the update law.

The architecture of the RNN is illustrated in Fig. 3. The input vector of the RNN $X = [x_1, x_2, x_3]^T$ represents the position, velocity and acceleration of the manipulators at each joint, respectively. Note that the calculation of position and velocity can be easily obtained, but that of the acceleration is not easy.

In this paper, as an approximation it is computed by differentiating the velocity using a first-order filter. Although this is not a desirable process due to possible side-effects such as increasing susceptibility noise, such measurements are successfully used in various adaptive control algorithms in some real-time applications [45].

B. Learning Rule for RNNs

The forward computation is described as follows:

Input layer

$$I_m(k) = x_m(k) \quad (m = 1, 2, 3). \tag{30}$$

Hidden layer

$$\begin{aligned} H_j(k) &= F_j\left(net_j(k)\right) \\ net_j(k) &= w_j^{(2)} H_j(k-1) + \sum_j w_{mj}^{(1)} x_m(k). \end{aligned} \tag{31}$$

Output layer

$$O(k) = N_i(\nu, \dot{\nu}, \ddot{\nu}, \omega) = \sum_j w_j^{(3)} H_j(k) \tag{32}$$

where $net_j(k)$ and $H_j(k)$ are the input and output of the hidden layer at the jth unit, respectively, $O(k)$ is the output of the RNN, namely, the approximation of inverse dynamics at each joint, $F_j(*)$ is sigmoid function, written as $F_j(*) = 1/(1+\exp^{(-*)})$, $w_{mj}^{(1)}$, $w_j^{(2)}$, and $w_j^{(3)}$ are the weights from input to hidden layer, last state to present state of hidden layer, hidden to output layer of the RNN, respectively.

The gradient-descent updating equations are

$$\frac{\partial E_i}{\partial w_j^{(3)}} = -\varepsilon_i(k)\frac{\partial O(k)}{\partial w_j^{(3)}} \tag{33}$$

$$\begin{aligned} \frac{\partial E_i}{\partial w_j^{(2)}} &= -\varepsilon_i \frac{\partial O(k)}{\partial H_j(k)} \frac{\partial H_j(k)}{\partial w_j^{(2)}} \\ &= -\varepsilon_i w_j^{(2)} \delta_j^{(i)}(k) \end{aligned} \tag{34}$$

$$\frac{\partial E_i}{\partial w_{mj}^{(1)}} = -\sum_j \left[\varepsilon_i \frac{\partial O(k)}{\partial H_j(k)}\right] \frac{\partial H_j(k)}{\partial w_{mj}^{(1)}} = -\sum_j \left[\varepsilon_i w_j^{(3)}\right] \beta_{mj}^{(i)}(k) \tag{35}$$

where subscript i denotes the neural network which approximates the ith joint of the manipulator

$$\begin{aligned}\delta_j^{(i)}(k) &= \frac{\partial H_j(k)}{\partial w_j^{(2)}} \\ &= F'\left(net_j(k)\right)\left[H_j(k-1) + w_j^{(2)}\delta_j^{(i)}(k-1)\right] \\ \beta_{mj}^{(i)}(k) &= \frac{\partial H_j(k)}{\partial w_{mj}^{(1)}} \\ &= F'\left(net_j(k)\right)\left[x_m(k) + w_j^{(2)}\beta_{mj}^{(i)}(k-1)\right].\end{aligned}$$

Then, the weights of the RNN can be updated according to the following expression:

$$w(k+1) = w(k) - \alpha \nabla E_i(w) \tag{36}$$

where $w = \left[w_{mj}^{(1)}, w_j^{(2)}, w_j^{(3)}\right]^T$, α is an adaptive learning rate.

Therefore, the connection weights are updated according to a simple first-order differential equation involving the parameter α, the current value obtained from a servo feedback loop, and the local response of the corresponding network. It is important to point out that the above development depends on the assumption that the RNN is capable of approximating the inverse dynamics of the controlled system. It is well known that neural network can be capable of approximating any reasonable function. However, as it will be made evident shortly, it is not sufficient for stability. Stability imposes additional requirement that the error in the approximation of the inverse dynamics has to remain bounded. Not surprisingly, it turns out that the condition on the boundedness of the approximation error can be satisfied by carefully choosing the network parameters, so as to guarantee a desired uniform approximation in a target set.

Note that the above update equations define a system of coupled nonlinear differential equations, this hampers a global stability analysis. However, local stability properties of closed loop system can be investigated.

IV. Stability and Convergence Analysis

The learning law (36) calls for a proper choice of the learning rate α. For a small value of α the convergence is guaranteed but the convergence speed is slow. On the other hand, if α is too big, the control system becomes unstable. This section will develops a guideline for selecting α properly, which will lead to an adaptive learning rate [46].

A discrete-type Lyapunov function can be given by

$$V_L(k) = \frac{1}{2}\varepsilon_i^2(k). \tag{37}$$

Thus, the change of the Lyapunov function due to the training process is obtained by

$$\Delta V_L(k) = V_L(k+1) - V_L(k) = \frac{1}{2}\left[\varepsilon_i^2(k+1) - \varepsilon_i^2(k)\right]. \tag{38}$$

The error difference due to the learning can be represented by

$$\varepsilon_i(k+1) = \varepsilon_i(k) + \Delta\varepsilon_i(k) = \varepsilon_i(k) + \left[\frac{\partial E_i}{\partial w}\right]^T \Delta w \tag{39}$$

where Δw represents a change in an arbitrary weight vector.

From the update law (33)–(35)

$$\Delta w = -\alpha\varepsilon_i(k)\frac{\partial E_i}{\partial w} = \alpha\varepsilon_i(k)\frac{\partial O(k)}{\partial w}. \tag{40}$$

Then, we have the following general convergence lemma [47]:

Lemma 1: Let α be the learning rate for the weights of the RNN and $z_{\max}$ be defined as $z_{\max} = \max\|z(k)\|^2$, where $z(k) = \partial O(k)/\partial w$, and $\|*\|$ is the usual Euclidean norm in R^n. The convergence is guaranteed if α is chosen as

$$0 < \alpha < \frac{2}{z_{\max}^2}. \tag{41}$$

Now, by using the lemma we can determine the learning rate α to guarantee the convergence of RNN learning.

Theorem 1: Let α_3 be the learning rate for the RNN weights $w^{(3)}$. The dynamic backpropagation algorithm converges if $0 < |w_j(3)| < 1 (j = 1, 2, \cdots, h)$ and the learning rate α_3 is chosen as

$$0 < \alpha_3 < \frac{2}{h} \tag{42}$$

where h is the number of the recurrent neurons in hidden layer.

Proof: From (33)

$$z(k) = \frac{\partial O(k)}{\partial w^{(3)}} = H(k). \tag{43}$$

where $H(k) = [h_1(k), h_2(k), \cdots, h_h(k)]^T$, and $h_j(k)$ is the output value of the jth neuron in the hidden layer. Since $0 < h_j(k) < 1$ (this can be verified from the nonlinear sigmoid function $F(*) = 1/(1+\exp^{(-*)})$. By the definition of the usual Euclidean norm in R^h, $\|z(k)\| < \sqrt{h}$ and $z_{\max}(k) = h$, Then, from Lemma, (42) can be obtained. □

Similarly, the learning rates α_2 and α_1 for the weights $w^{(2)}$ and $w^{(1)}$ can be chosen according to the following expressions, respectively.

$$0 < \alpha_2 < \frac{1}{h}\left[\frac{1}{w_{max}^{(2)}}\right]^2 \tag{44}$$

$$0 < \alpha_{13} < \frac{1}{h+3}\left[\frac{1}{w_{\max}^{(1)} x_{\max}}\right]^2 \tag{45}$$

where $w_{\max}^2 = \max\|w^{(2)}(k)\|$, $w_{\max}^1 = max\|w^{(1)}(k)\|$, and $\|*\|$ is the sup-norm. The detailed proof can be referred to the Appendix of [47].

V. Force Tracking and Composite Control

A. Force Tracking

Now, we consider the relation of the constrained force f_n, the axial force Q, and the Lagrange multiplier λ. From (1) and (2), the constrained condition can be written as

$$\begin{aligned}&\Omega(\theta_1,\theta_2,w_E)\\&\quad=\Omega(X_p,Y_p)\\&\quad=\Omega\left(L_1\cos\theta_1+L_2\cos(\theta_1+\theta_2)-w_E\sin(\theta_1+\theta_2)\right.\\&\qquad\left.L_1\sin\theta_1+L_2\sin(\theta_1+\theta_2)+w_E\cos(\theta_1+\theta_2)\right)=0.\end{aligned}$$

The reaction force from the constraint surface is given by

$$f_n n=\lambda\begin{bmatrix}b(\theta_1,\theta_2,w_E)\\c(\theta_1,\theta_2,w_E)\end{bmatrix}\tag{46}$$

where $b(\theta_1,\theta_2,w_E)=\partial\Omega(X_p,Y_p)/\partial X$, $c(\theta_1,\theta_2,w_E)=\partial\Omega(X_p,Y_p)/\partial Y$.

Because the axial force Q is the orthogonal projection of the reaction force $f_n n$ on the unit vector $-i_2$ as shown in Fig. 1, it can be represented as

$$Q=-f_n i_2^T n.\tag{47}$$

Using (46) and (47), we have

$$Q=-\lambda\left[b(\theta_1,\theta_2,w_E)\cos(\theta_1+\theta_2)+c(\theta_1,\theta_2,w_E)\sin(\theta_1,\theta_2)\right].\tag{48}$$

As we assume that the deviation w_E from deformation is small compared with L_1 and L_2, the relation in (48) can be approximated as

$$\lambda=\frac{-Q}{\bar{b}(\theta_1,\theta_2)\cos(\theta_1+\theta_2)+\bar{c}(\theta_1,\theta_2)\sin(\theta_1+\theta_2)}\tag{49}$$

where $\bar{b}(\theta_1,\theta_2)\cong b(\theta_1,\theta_2,w_E)$, and $\bar{c}(\theta_1,\theta_2)\cong c(\theta_1,\theta_2,w_E)$. Because the axial force can be measured by using a force sensor mounted at the tip of the second link, it is possible to obtain the value of the Lagrange multiplier λ from this equation. The unit vector of the constraint surface is given as

$$n=\frac{1}{\sqrt{\bar{b}^2(\theta_1,\theta_2)+\bar{c}^2(\theta_1,\theta_2)}}\begin{bmatrix}\bar{b}(\theta_1,\theta_2)\\\bar{c}(\theta_1,\theta_2)\end{bmatrix}.\tag{50}$$

From (46) and (50), we have

$$f_n=\lambda\sqrt{\bar{b}^2(\theta_1,\theta_2)+\bar{c}^2(\theta_1,\theta_2)}.\tag{51}$$

Therefore, combining with (17), the contact force of the end-effector with the constraint can be obtained.

B. Composite Control

The control objective is position and force trajectory tracking. Therefore, we will define $\nu(t)=[\nu_{1d}(t),\nu_{2d}(t)]^T$, as the desired trajectory for the reduced-order variable ν defined in (12). Note that $\nu_{2d}=\dot{\nu}_{2d}=\ddot{\nu}_{2d}=0$, for the end-effector to move on the constraint all along during the whole execution. Likewise, we define the desired Lagrange multiplier to be λ_d. Then, the force tracking problem involves requiring that the generalized multiplier λ can track the desired λ_d. Therefore, we define the force tracking error as

$$e_\lambda=\lambda_d-\lambda.\tag{52}$$

Given the above definition, and combining with (14) and (22), the composite control law is given as follows:

$$\tau=T_r^{-T}\begin{bmatrix}k_{p1}e_1+k_{d1}\dot{e_1}+N_1(*)\\e_\lambda\end{bmatrix}\tag{53}$$

where $e_1=\nu_{1d}-\nu_1(=\theta_{1d}-\theta_{1a})$, similarly for $\dot{e_1}$, k_{1p}, k_{1d} are the corresponding constant gains, $N_1(*)$ is the output of the corresponding RNN.

VI. Simulation Results

We begin describing a series of simulation tests to study the adaptive behaviors of the proposed composite control scheme based on the RNN model and its effectiveness for controlling constrained flexible robot manipulators. These tests are performed on the model of a two-DOF flexible manipulator. We will focus our attention on the tracking accuracy of the manipulators in both position and force, and deflection of the flexible link of the robotic system. As to the selection of the number of modes, for the flexible manipulator system there will be an infinite number of modes of vibration of the flexible manipulator due to the distributed nature of the system. However, in practice, the contribution of higher modes to the overall movement is found to be negligible while the computational complexity is cubic in the number of modes [48]. Therefore, a reduced-order model incorporating the lower modes is preferred. Here, we choose the first two modes to describe the vibration of the flexible manipulator.

In our simulations, the constrained rigid surface is taken as a straight line $X+Y=1$. The specific constrained condition corresponding (1) is expressed as

$$\begin{aligned}\Omega(X,Y)&=\Omega(\theta_1,\theta_2,w_E)\\&=L_1\sin\theta_1+L_2\sin(\theta_1+\theta_2)-w_E\cos(\theta_1+\theta_2)\\&\quad+L_1\cos\theta_1+L_2\cos(\theta_1+\theta_2)\\&\quad+w_E\sin(\theta_1+\theta_2)-1=0.\end{aligned}$$

For the end-effector of the manipulator to move on the constraint all the time, θ_1 and θ_2 are not independent. Therefore, the angle θ_2 can be expressed as a function of θ_1

$$\begin{aligned}\theta_2&=\Omega(\theta_1)\\&=\pi-\sin^{-1}\frac{1-L_1(\cos\theta_1+\sin\theta_1)}{\sqrt{(L_1+w_E)^2+(L_1-w_E)^2}}\\&\quad-\theta_1-\tan^{-1}\frac{L_1-w_E}{L_1+w_E}.\end{aligned}$$

The physical parameters are listed in Table I. the desired position trajectory is given as $\nu_1=0.4+0.3\sin(0.04\pi t)$, and the desired contact force is kept as $f_d=10\,\mathrm{N}$ during the whole process. First, to compare the performance of the PID with the proposed control law, the simulation is carried out, the results are shown in Figs. 4 and 5, respectively. In which, (a)–(d) are the position

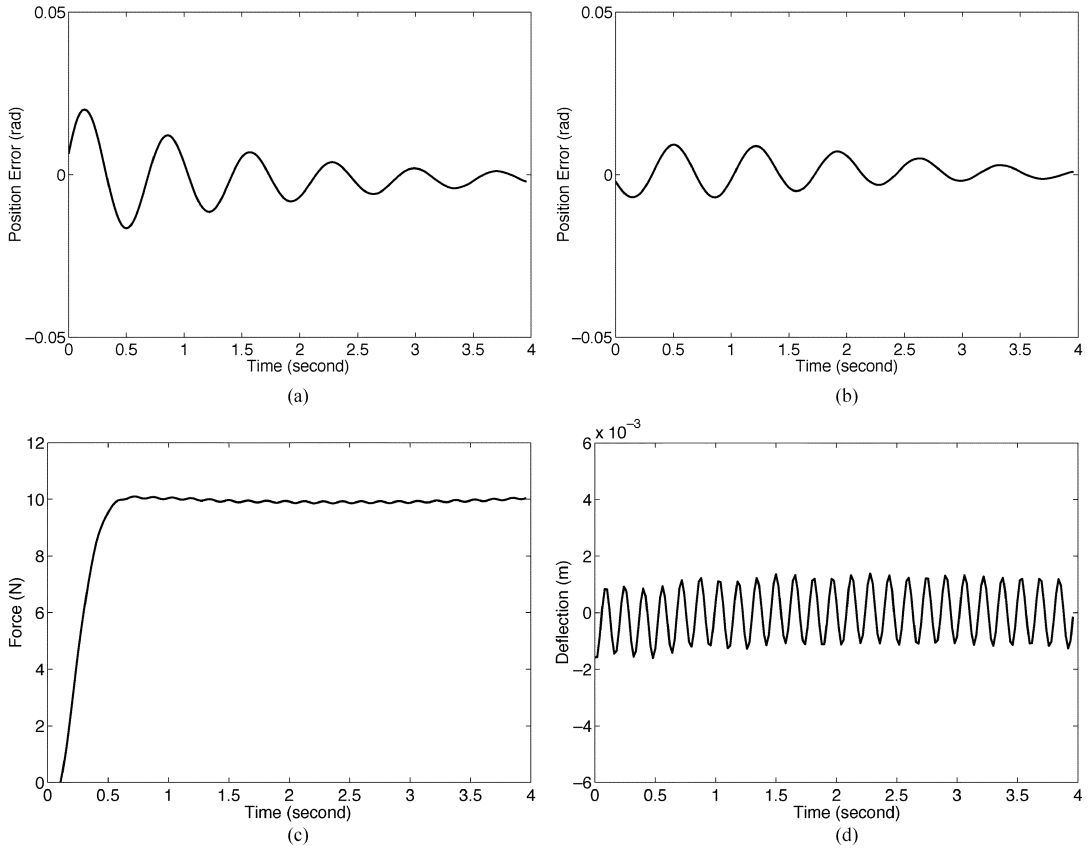

Fig. 4. Transient responses with a PID controller.

TABLE I
PARAMETERS OF THE FLEXIBLE MANIPULATOR SYSTEM

Physical parameters	Link 1	Link 2
Length	$L_1 = 0.45m$	$L_2 = 0.38m$
Moment of inertia	$J_1 = 0.3kg \cdot m^2$	$J_h = 0.05kg \cdot m^2$
Density		$\rho = 0.17kg \cdot m^2$
Elastic modulus		$EI = 1.0N \cdot m^2$
Mass		$M_h = 0.2kg$

tracking errors of joints 1 and 2, contact force tracking, and vibration, respectively (similarly hereinafter). Using a PID controller, it appears an attenuation signal with amplitude of 0.02 rad, and after about 4 s the steady state can be reached. While using the proposed controller, the position error of joint 1 can be controlled within ± 0.0001 rad, and the error for joint 2 is within ± 0.0006 rad. Force error can be controlled within ± 0.2 N for both controllers. Though it appears overshoot of about 0.4 N by the proposed control law, the rising time is shortened greatly. The deflection of the flexible manipulator is also considered. The simulation results are encouraging. It indicates that maximum deflection is 0.0005 m, only one third of that by using a PID controller. It can be concluded that the performance of the control system can be improved greatly by using the proposed control law.

To investigate the adaptive capability of the proposed controller, simulations were also carried out by increasing the concentrated mass m_h and moment of inertia J_1, while keeping other physical and controller parameters unchanged. The simulation results are depicted in Figs. 6 and 7. It can be seen that the maximum position tracking errors of joints 1 and 2 increase to 0.0015 rad, 0.5% of the amplitude of the input signal, while the contact force and the deflection is neatly unchanged for both two cases. Even though the position tracking errors increase much more than the previous case, it can still meet the requirement to normal industrial applications.

Similarly, to see the performance of the proposed controller when the velocity of the end-effector of the flexible manipulator is increased, simulations were conducted. Fig. 8 illustrates the results with doubled velocity compared to the previous case. From Fig. 8 it is shown that the position tracking errors increase to 0.002 rad and 0.0038 rad, respectively. However, they are larger than the previous case, only 0.45% and 0.85% of the input signals, it still keeps high accuracy. The contact force is nearly unchanged. The maximum vibration increases to 0.0015 m. Based on this, it can be concluded that it is not favorable for weakening the vibration of the flexible manipulator increasing velocity. So, in order to make the deflection smaller, it is better not to pursuing higher velocity.

Fig. 9 shows the simulation results when an initial position error exists, the result is obtained when 0.02 rad initial position error, 5.7% of the amplitude of the input signal, is given. It indicates that the position trajectory tracking and the contact force are almost not affected by the initial position error. But, the vibration increases from 0.0005 m to 0.0015 m, tripled the previous case. In short, from Figs. 8 and 9, it is demonstrated that it is desirable to avoid higher velocity and initial position error in order to obtain excellent performance, especially the small deflection.

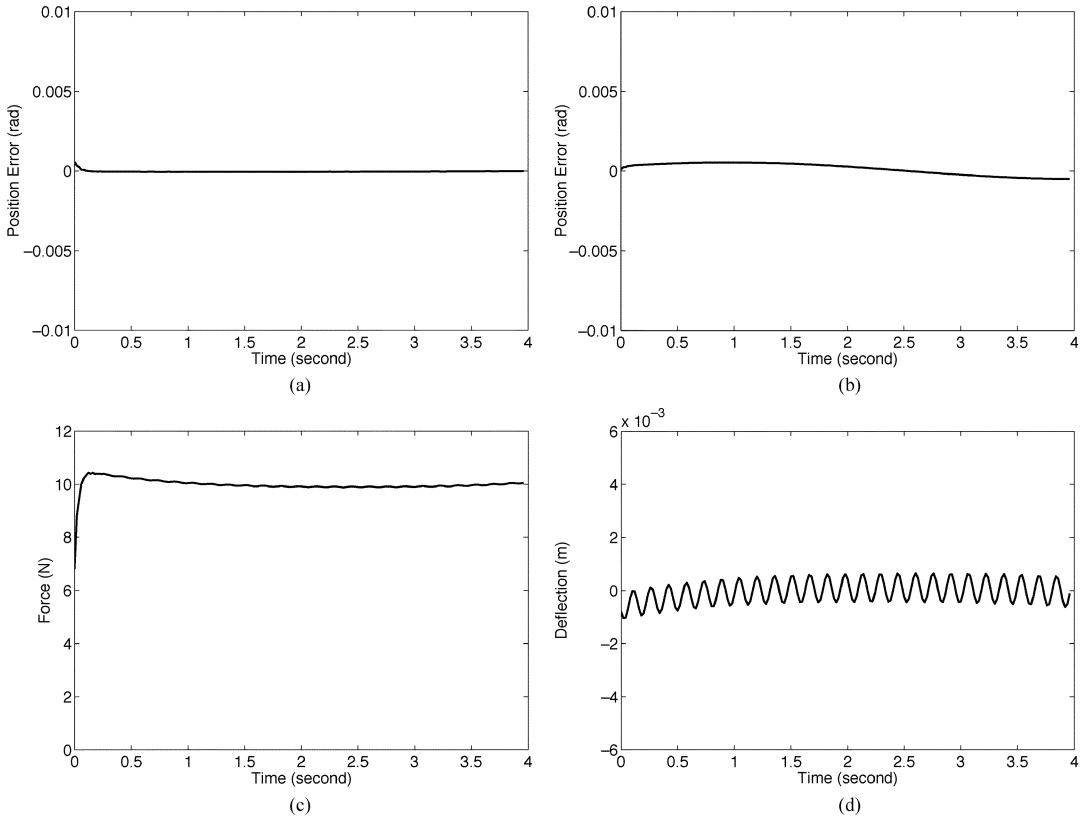

Fig. 5. Transient responses with proposed controller.

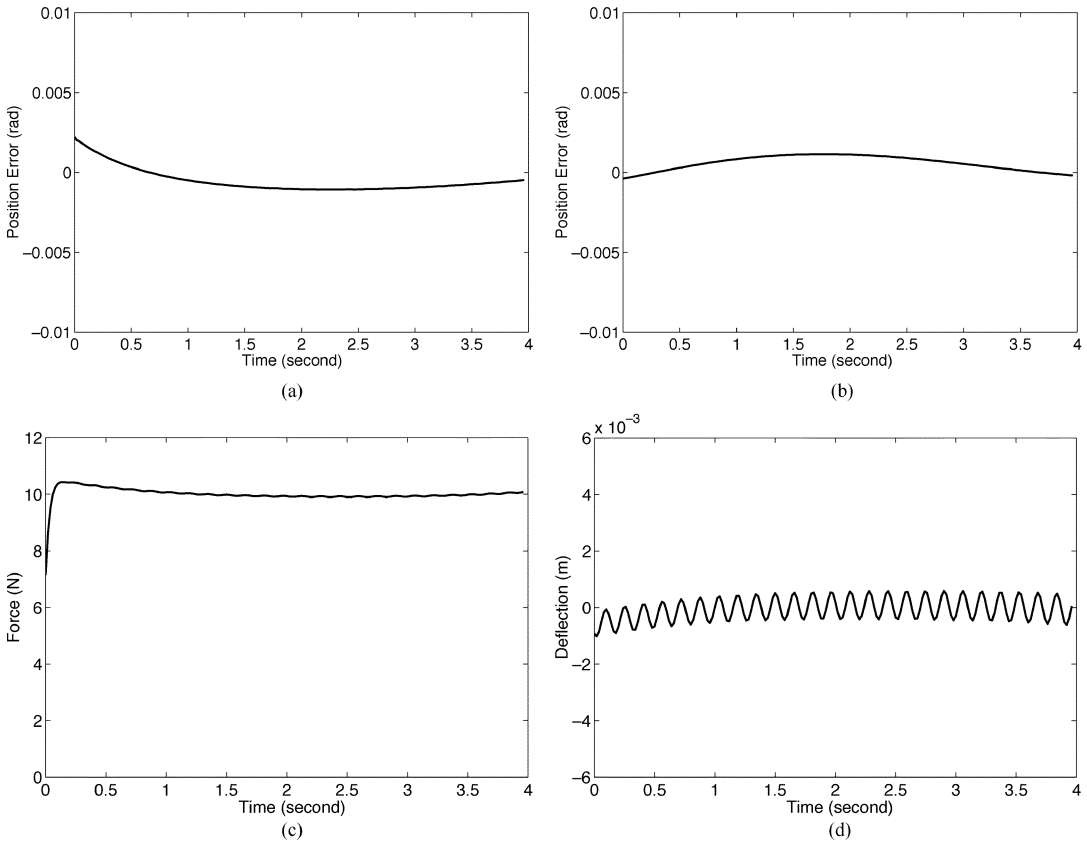

Fig. 6. Transient responses with increased mass.

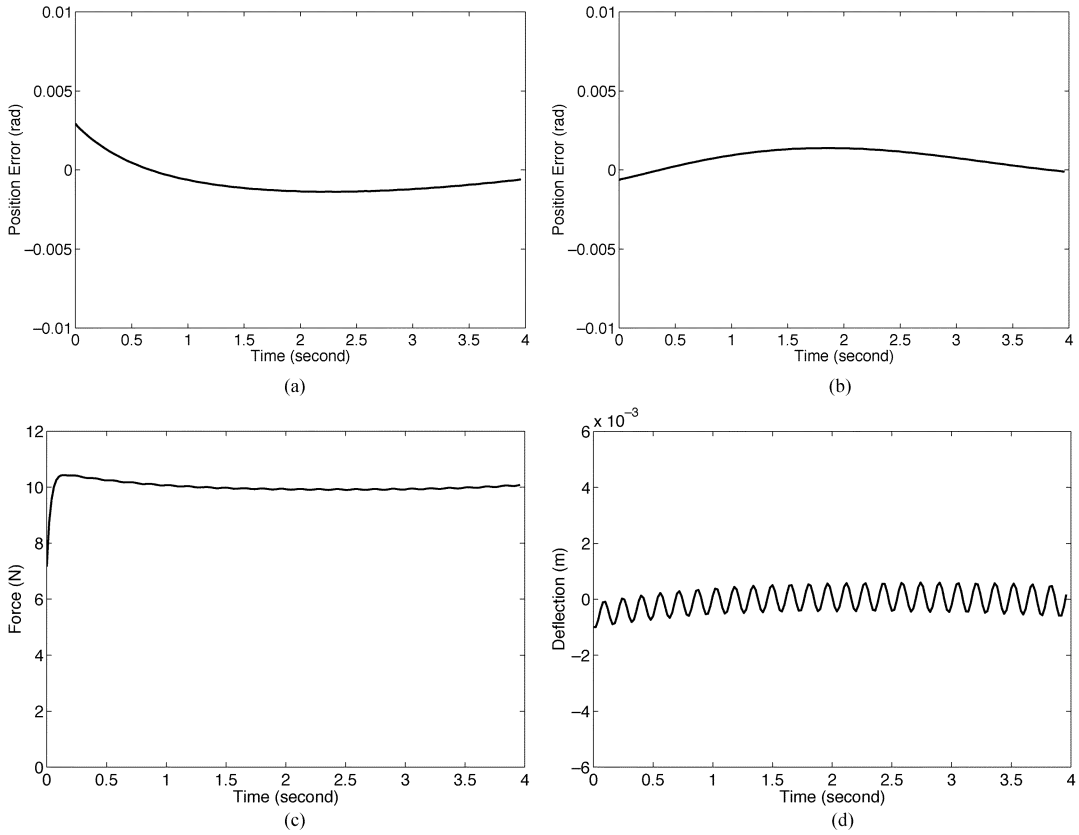

Fig. 7. Transient responses with increased moment of inertia.

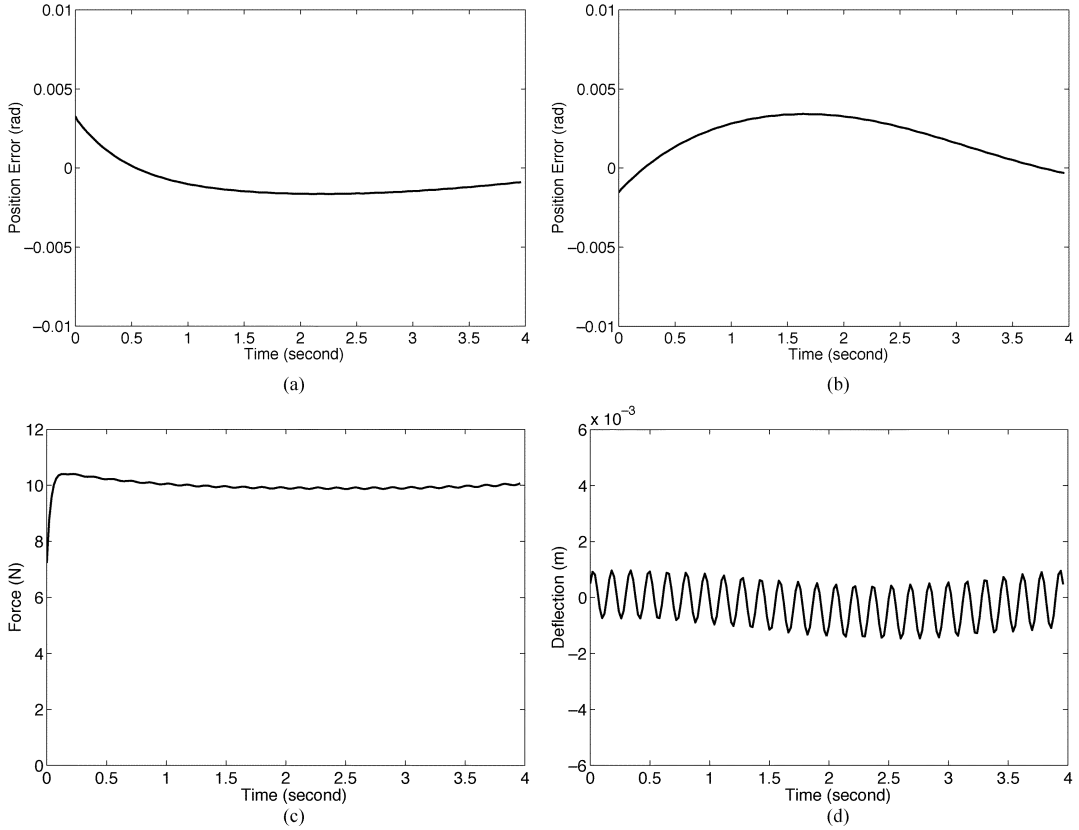

Fig. 8. Transient response with increased velocity.

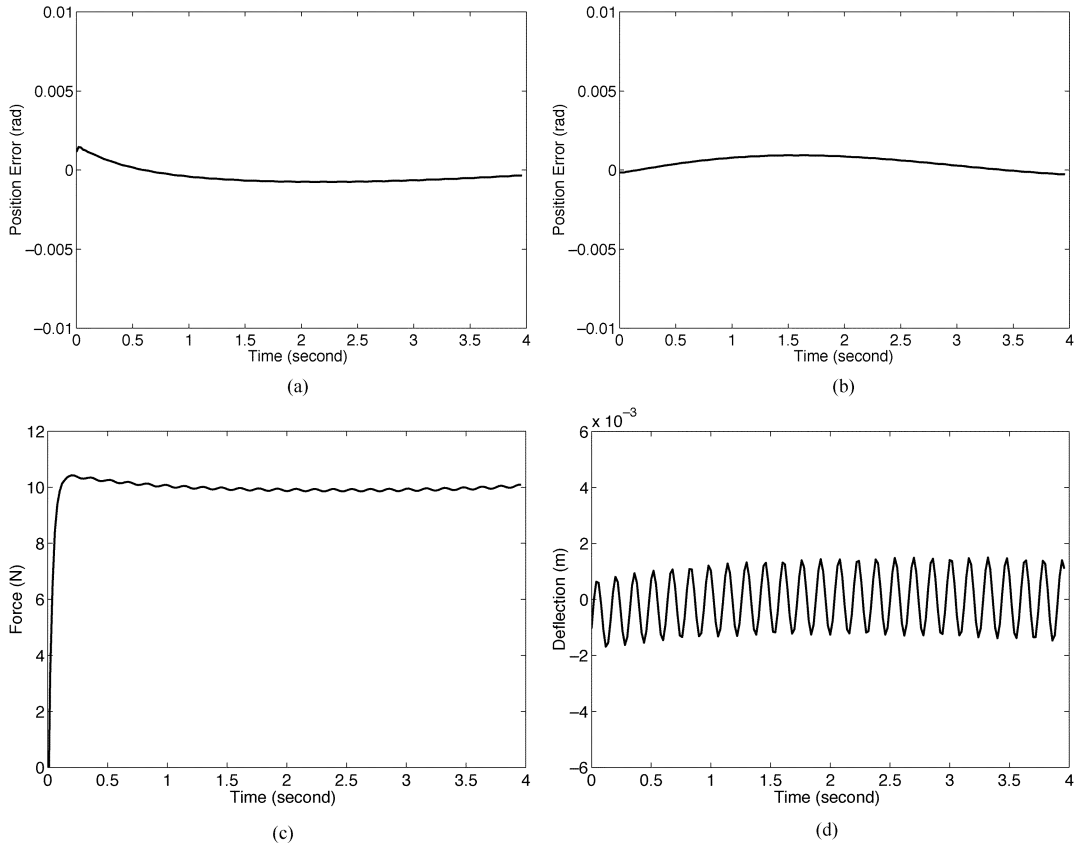

Fig. 9. Transient response with initial position error.

VII. Concluding Remarks

We have investigated the hybrid position/force control of constrained flexible manipulators. A new control law based on RNNs is proposed to approximate the dynamics of manipulators and compensate for more uncertainties. Local stability and convergence are analyzed. Based on the proposed control law, a composite controller is designed. Comparison between the traditional PID control method and the proposed control law is investigated. Adaptive capability is tested when the physical parameters and external variables undergo changing. External uncertainties such as initial position errors are also considered. The simulation results demonstrate that the proposed neural network approach has excellent performance compared with the PID method.

References

[1] S. Jung and T. C. Hsia, "Neural network impedance force control of robot manipulators," *IEEE Trans. Ind. Electron.*, vol. 45, pp. 451–461, June 1998.

[2] ——, "Robust neural force control scheme under uncertainties in robot dynamics and unknown environment," *IEEE. Trans. Ind. Electron.*, vol. 47, pp. 403–412, Apr. 2000.

[3] A. Yazdizadeh, K. Khorasani, and R. V. Patel, "Identification of a two-link flexible manipulator using adaptive time delay neural networks," *IEEE Trans. Syst., Man, Cybern. B*, vol. 30, pp. 165–172, Feb. 2000.

[4] L.-C. Lin and T.-W. Yih, "Rigid model-based neural network control of flexible-link manipulators," *IEEE Trans. Robot. Automat.*, vol. 12, pp. 595–602, Aug. 1996.

[5] Z. Su and K. Khorasani, "A neural-network-based controller for a single-link flexible manipulator using the inverse dynamics approach," *IEEE Trans. Ind. Electron.*, vol. 48, pp. 1074–1086, Apr. 2001.

[6] J. M. Martins, Z. Mohamed, M. O. Tokhi, J. Sa da Costa, and M. A. Botto, "Approaches for dynamic modeling of flexible manipulator systems," *Inst. Elec. Eng. Proc. Contr. Theory and Applications*, vol. 150, no. 4, pp. 401–411, 2003.

[7] M. Bai, D. H. Zhou, and H. Schwarz, "Adaptive augmented state feedback control for an experimental plannar two-link flexible manipulator," *IEEE Trans. Robot. Automat.*, vol. 14, pp. 940–950, Dec. 1998.

[8] H. Yang, H. Krishman, and M. H. Ang Jr., "Tip-trajectory tracking control of single-link flexible robots by output redefinition," *Inst. Elec. Eng. Proc. Contr. Theory Applicat.*, vol. 147, no. 6, pp. 580–587, 2000.

[9] J. B. Mbede, X. Huang, and M. Wang, "Robust neural-fuzzy sensor-based motion control among dynamic obstacles for robot manipulators," *IEEE Trans. Fuzzy Syst.*, vol. 11, pp. 249–260, Apr. 2003.

[10] H. D. Patino, R. Carelli, and B. R. Kuchen, "Neural networks for advanced control of robot manipulators," *IEEE Trans. Neural Networks*, vol. 13, pp. 343–354, March 2002.

[11] L. Behera, M. Gopal, and S. Chaudhury, "On adaptive trajectory tracking of a robot manipulator using inversion of its neural emulator," *IEEE Trans. Neural Networks*, vol. 7, pp. 1401–1414, Nov. 1996.

[12] R. Carelli, E. F. Camacho, and D. Patino, "A neural network based feedforward adaptive controller for robots," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-25, pp. 1281–1288, Sept. 1995.

[13] Y. Gao, M. J. Er, and S. Yang, "Adaptive control of robot manipulators using fuzzy neural networks," *IEEE Trans. Ind. Electron.*, vol. 48, pp. 1274–1278, Dec. 2001.

[14] S. S. Ge, C. C. Hang, and L. C. Woon, "Adaptive neural network control of robot manipulators in task space," *IEEE Trans. Ind. Electron.*, vol. 44, pp. 746–752, Dec. 1997.

[15] W. Cheng and J. T.-Y. Wen, "A two-time-scale neural controller for the tracking control of rigid manipulators," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-24, pp. 991–1000, July 1994.

[16] S. S. Ge, T. H. Lee, and C. J. Harris, *Adaptive Neural Network Control of Robotic Manipulators*, Singapore: World Scientific, 1998.

[17] F. L. Lewis, S. Jagannathan, and A. Yesildireh, *Neural Network Control of Robot Manipulators and Nonlinear Systems*. New York: Taylor & Francis, 1999.

[18] M. J. Er and Y. Gao, "Roust adaptive control of robot manipulators using generalized fuzzy neural networks," *IEEE Trans. Ind. Electron.*, vol. 50, pp. 620–628, June 2003.
[19] F. C. Sun, Z. Q. Sun, and P.-Y. Woo, "Neural network-based adaptive controller design of robotic manipulators with an observer," *IEEE Trans. Neural Networks*, vol. 12, pp. 54–67, Jan. 2001.
[20] Y. H. Kim and F. L. Lewis, "Neural network output feedback control of robot manipulators," *IEEE. Trans. Robot. Automat.*, vol. 15, pp. 301–309, Apr. 1999.
[21] M.-J. Lee and Y.-K. Choi, "An adaptive control method for robot manipulators using radial basis function networks," in *Proc. IEEE Int. Symp. Industrial Electronics*, 2001, pp. 1827–1832.
[22] A. Karakasoglu and M. K. Sundareshan, "A recurrent neural network-based adaptive variable structure model-following control of robotic manipulators," *Automatica*, vol. 31, no. 19, pp. 1495–1507, 1995.
[23] B. Song and A. J. Koivo, "Nonlinear predictive control with application to manipulator with flexible forearm," *IEEE Trans. Ind. Electron.*, vol. 46, pp. 923–932, Oct. 1999.
[24] L.-C. Lin and T.-W. Yih, "Rigid model-based neural network control of flexible-link manipulators," *IEEE Trans. Robot. Automat.*, vol. 12, pp. 595–602, Aug. 1996.
[25] K. Kiguchi and T. Fukuda, "Position/force control of robot manipulators for geometrically unknown objects using neural networks," *IEEE Trans. Ind. Electron.*, vol. 47, pp. 641–649, June 2000.
[26] F. C. Sun, Z. Q. Sun, R. J. Zhang, and Y. B. Chen, "Discrete-time tracking control of robotic manipulators based on dynamic inversion using dynamic neural networks," in *Proc. IEEE Int. Symp. Intelligent Control*, 2000, pp. 333–338.
[27] S. H. Hu, M. H. Ang Jr, and H. Krishnan, "Neural network controller for constrained robot manipulators," in *Proc. IEEE Int. Conf. Robotics Automation*, 2000, pp. 1906–1911.
[28] S. Jung and T. C. Hsia, "On an effective approach of Cartesian space neural network control for robot manipulator," *Robotica*, vol. 15, pp. 305–312, 1997.
[29] ——, "A study of neural network control of robot manipulators," *Robotica*, vol. 14, pp. 7–15, 1996.
[30] D. Wang and M. Vidyasagar, "Control of a class of manipulator with a single flexible link—part 1: feedback linearization," *Trans. ASME, J. Dyn. Syst., Meas., Control*, vol. 113, pp. 655–661, 1991.
[31] B. C. Chiou and M. Shahinpoor, "Dynamic stability analysis of a one-link force-controlled flexible manipulator," *J. Robot. Syst.*, vol. 5, no. 5, pp. 443–451, 1988.
[32] F. L. Hu and A. G. Ulsoy, "Force and motion control of a constrained flexible robot arm," *Trans. ASME, J. Dyn. Syst., Meas., Control*, vol. 116, pp. 336–343, 1994.
[33] M. M. Bridges, J. Cai, D. M. Dawson, and M. T. Grabbe, "Experimental results for a robust position and force controller implemented on a direct drive robot," *Robotica*, vol. 13, pp. 11–18, 1995.
[34] C. C. Cheah, D. W. Wang, and Y. C. Soh, "Learning control of motion and force for constrained robotic manipulators," *Int. J. Robot. Automat.*, vol. 10, no. 3, pp. 79–88, 1995.
[35] Y. Zhang and J. Wang, "A dual neural network for constrained torque optimization of kinematically redundant manipulators," *IEEE Trans. Syst., Man, Cybern. B*, vol. 32, no. 5, pp. 654–662, 2002.
[36] J. Wang, Q. Hu, and D. Jiang, "A Lagrangian network for kinematic control of redundant robot manipulators," *IEEE Trans. Neural Networks*, vol. 10, pp. 1123–1132, Sept. 1999.
[37] Y. H. Kim and F. L. Lewis, "Optimal design of CMAC neural-network controller for robot manipulators," *IEEE Trans. Syst., Man, Cybern. C*, vol. 30, Feb. 2000.
[38] Y. Xia and J. Wang, "A dual neural network for kinematic control of redundant robot manipulators," *IEEE Trans. Syst., Man Cybern. B*, vol. 31, pp. 147–154, Feb. 2001.
[39] C.-J. Chieu and L.-C. Fu, "A neural network based learning controller for robot manipulators," in *Proc. 39th IEEE Conf. Decision Control*, 2000, pp. 1748–1753.
[40] Y. Zhang, J. Wang, and Y. Xu, "A dual neural network for bi-criteria kinematic control redundant manipulators," *IEEE Trans. Robot. Automat.*, vol. 18, pp. 923–931, Dec. 2002.
[41] J. I. Arciniegas, A. H. Eltimsahy, and K. J. Cios, "Neural-networks-based adaptive control of flexible robotic arms," *Neurocomputing*, vol. 17, pp. 141–157, 1997.
[42] W. S. Tang and J. Wang, "Two recurrent neural networks for joint torque optimization of kinematically redundant manipulators," *IEEE Trans. Syst., Man, Cybern. B*, vol. 30, pp. 120–128, Feb. 2000.
[43] ——, "A recurrent neural network for minimum infinity-norm kinematic control of redundant manipulators with an improved problem formulation and reduced architecture complexity," *IEEE Trans. Syst., Man, Cybern. B*, vol. 31, pp. 98–105, Feb. 2001.
[44] M. K. Ciliz and C. Isik, "On-line learning control of manipulators based on artificial neural network models," *Robotica*, vol. 15, pp. 293–304, 1997.
[45] D. H. Rao and M. M. Gupta, "Neuro-fuzzy controller for control and robotics applications," *Eng. Applicat. Artif. Intell.*, vol. 7, no. 5, pp. 479–491, 1994.
[46] B. Siciliano and W. J. Book, "A singular perturbation approach to control of lightweight flexible manipulators," *Int. J. Robot. Res.*, vol. 7, no. 4, pp. 79–90, 1988.
[47] C.-C. Ku and K. Y. Lee, "Diagonal recurrent neural networks for dynamic systems control," *IEEE Trans. Neural Networks*, vol. 6, pp. 144–155, Jan. 1995.
[48] A. Jain and G. Rodrigues, "Spatially recursive dynamics for flexible manipulators," in *Proc. IEEE Int. Conf. Robotics Automation*, 1991, pp. 2350–2355.

Lianfang Tian received the B.S. and M.S. degrees in mechanical engineering from the Shandong University of Technology, Jinan, China, and the Ph.D. degree in mechanical and electrical engineering from Harbin Institute of Technology, Harbin, China, in 1991, 1994, and 1997, respectively.

He was a Postdoctor Research Fellow in automatic control engineering at South China University of Technology, Guangzhou, China, from 1998 to 1999. In 2000, he was a Research Fellow in the Department of Mechanical Engineering, University of California, Riverside. Since 2001, he has been a Research Fellow in the Department of Bioengineering at the University of Pittsburgh, Pittsburgh, PA. His research involves several areas including robotics, biomechanics, electro-mechanical system design and control, intelligent control, and electro-hydraulic servo control systems. He is Reviewer for several international journals such as *Robotica*, *Information Science* and *Journal of Intelligent and Fuzzy Systems*.

Dr. Tian is Reviewer for IEEE TRANSACTIONS. ON SYSTEMS, MAN, AND CYBERNETICS-PART B.

Jun Wang (S'89–M'90–SM'93) received the B.S. degree in electrical engineering and the M.S. degree in systems engineering from Dalian University of Technology, Dalian, China, and the Ph.D. degree in systems engineering from Case Western Reserve University, Cleveland, OH.

He is a Professor of automation and computer-aided engineering at the Chinese University of Hong Kong, Hong Kong. Prior to coming to Hong Kong in 1995, he was an Associate Professor at the University of North Dakota, Grand Forks. His current research interests include neural networks and their engineering applications.

Dr. Wang is an Associate Editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS and IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: PARTS B AND C.

Zongyuan Mao received the B.S. degree in shipbuilding engineering from Dalian University of Technology, China in 1962.

He is Professor in the College of Automation Science and Engineering at South China University of Technology, Guangzhou, China. His current research interests include intelligent control and their engineering applications. He is on the editorial board of the Chinese Journal *Control Theory and Applications*.