J Control Theory Appl 2013 11 (2) 222–229
DOI 10.1007/s11768-013-1240-x

Robust dynamic surface control of flexible joint robots using recurrent neural networks

Zhiqiang MIAO†, Yaonan WANG

College of Electrical and Information Engineering, Hunan University, Changsha Hunan 410082, China

Abstract: A robust neuro-adaptive controller for uncertain flexible joint robots is presented. This control scheme integrates H-infinity disturbance attenuation design and recurrent neural network adaptive control technique into the dynamic surface control framework. Two recurrent neural networks are used to adaptively learn the uncertain functions in a flexible joint robot. Then, the effects of approximation error and filter error on the tracking performance are attenuated to a prescribed level by the embedded H-infinity controller, so that the desired H-infinity tracking performance can be achieved. Finally, simulation results verify the effectiveness of the proposed control scheme.

Keywords: Dynamic surface control; Flexible joint robots; Robust H-infinity control; Recurrent neural network

1 Introduction

Over the past few decades, there has been much research on the tracking control of flexible joint robots. Experimental evidence has indicated that the flexibility of the joints should be taken into account in both modeling and control if high performance is to be achieved. Several control techniques have been proposed for flexible joint robot manipulators. Among them, the feedback linearization technique [1–2] relied on finding a diffeomorphic transformation and a static state feedback controller that render the system linear. The method, however, depended not only on the precise estimate of the system parameters, but also on the measurement of joint acceleration and jerk. Then, the singular perturbation method using integral manifold [3–5] was developed to overcome these limitations. Under the assumption of weak joint elasticity, singular perturbation modeling together with composite control design was used to improve dynamic performance [6]. The integral manifold approach is, however, only applicable to the weak joint elasticity case; and its performance degrades when the manipulator joints become significantly flexible.

The backstepping technique [7–10], in particular, has been widely used to design controllers for flexible joint robots. Oh and Lee [8] designed a controller for flexible joint robot manipulators by backstepping design approach. Bridges et al. [9] surveyed the backstepping control approach for flexible joint robots. Though such backstepping approach solve the control problem, a major problem of the backstepping technique is that certain unknown nonlinear function must be 'linear in the unknown parameters' and tedious analysis is needed to determine 'regression matrices' [11]. Therefore, several approaches have been presented for the adaptive control of uncertain nonlinear systems by incorporating the backstepping technique into the existing neural network based adaptive control design framework to overcome these problems [11–15]. Kwan et al. [11] developed a robust controller for flexible joint robots by using backstepping technique and neural networks. However, the backstepping algorithm has the problem of 'explosion of complexity' that is caused by the repeated differentiations of virtual controllers [16–17]. Swaroop et al. [16] proposed a dynamic surface control (DSC) technique adding a first-order low-pass filter at each step of the backstepping design procedure to eliminate this problem. The DSC technique was first combined with the neural adaptive scheme in [18] to handle nonlinear systems with arbitrary uncertainty. In [20–21], a new scheme combination of the adaptive dynamic surface control technique and the neural networks for the robust control of flexible-joint robots with model uncertainties was proposed.

In the past decade, great progress has been made in the study of using neural networks to control uncertain nonlinear systems. Extensive works demonstrate that adaptive neural control is particularly suitable for controlling highly uncertain, nonlinear, and complex systems [22]. In these neuro-adaptive control schemes, the neural network is used to compensate the effects of nonlinearity and system uncertainties, and so the stability, convergence and robustness of the control system can be improved [23]. According to the structure, the neural networks can be mainly classified as feedforward neural networks (FNNs) and recurrent neural networks (RNNs). It is well known that FNN is capable of approximating arbitrary continuous function closely. However, FNN is a static mapping and unable to represent

Received 9 December 2011; revised 3 June 2012.
†Corresponding author. E-mail: miaozhiqiang33@126.com. Tel.: +86-13786151828.
This work was supported by the National Natural Science Foundation of China (Nos. 60835004, 61175075), and the Hunan Provincial Innovation Foundation for Postgraduate (No. CX2012B147).

a dynamic mapping without the aid of tapped delays [24]. On the other hand, The RNNs, which comprise both feedforward and feedback connections, have superior capabilities than the FNNs. Since the RNN has a feedback loop, it can capture the past information of the network and adapt rapidly to sudden changes of the control environment [25]. The RNNs have the ability to deal with time-varying input or output though their own natural temporal operation [26]. For this ability, the structure of the neural network is simplified. Due to its dynamic characteristic and relative simple structure, the RNN is a useful tool in real-time application.

As in practical application, strong robustness is always an important property that a good controller should achieve. In [20], a robust recurrent neural network controller for flexible joint robots is derived by assuming that the bound of residual approximation error is known. To overcome this limitation, the issue of tracking with disturbance attenuation for uncertain systems has been studied in the nonlinear H_∞ setting [27–33]. In these control schemes, the controllers are generally composed of two main components. One is an adaptive neural network system that is used to approximate an ideal control law [27–28] or system uncertainties [29–32]. The other is a robust compensator that is designed to attenuate the effect of approximation error to a prescribed level so that the H_∞ tracking performance can be yield. In this paper, a robust neuro-adaptive controller for uncertain flexible joint robots is presented. This control scheme integrates H_∞ disturbance attenuation design and the recurrent neural network adaptive control technique into the dynamic surface control framework. First, the function approximation capacity of RNN is exploited for adaptively learning those uncertain dynamics in a flexible joint robot. Second, the effects on the tracking performance of the approximation error and filter error are then attenuated to a prescribed level by the embedded H_∞ controller. Finally, simulation results for a single link flexible joint robot verify the good tracking performance of the proposed control scheme.

This paper is organized as follows. The model and basic properties of flexible joint robot systems with uncertainties is introduced in Section 2. The RNN structure is presented in Section 3. In Section 4, a RNN-based dynamic surface controller for uncertain flexible joint robots is designed; in addition, the stability and robustness of the proposed control system are analyzed based on the Lyapunov stability theorem and H_∞ control theory. Simulation results are discussed in Section 5. Finally, conclusions are drawn in Section 6.

2 Flexible joint robots model and properties

In general, the model for an n-link flexible joint robot is given by

$$M(q)\ddot{q}+V(q,\dot{q})\dot{q}+G(q)+F(\dot{q})+Kq+T_{\mathrm{L}}=Kq_{\mathrm{f}}, \quad (1)$$

$$J\ddot{q}_{\mathrm{f}}+B\dot{q}_{\mathrm{f}}+K(q_{\mathrm{f}}-q)+T_{\mathrm{E}}=u, \quad (2)$$

where q, $\dot{q}$, $\ddot{q}$ denote the link position, velocity, and acceleration vectors, respectively, $M(q)\in\mathbb{R}^{n\times n}$ is the inertia matrix, $V(q,\dot{q})\in\mathbb{R}^{n\times n}$ denotes the centripetal coriolis matrix, $G(q)\in\mathbb{R}^n$ is the gravity vector, and $F(\dot{q})\in\mathbb{R}^n$ represents the friction term, and $T_{\mathrm{L}}\in\mathbb{R}^n$ the additive bounded disturbance. q_{f}, $\dot{q}_{\mathrm{f}}$, $\ddot{q}_{\mathrm{f}}$ denote the actuator position, velocity, and acceleration vectors, respectively. The constant positive definite diagonal matrices $K\in\mathbb{R}^{n\times n}$, $J\in\mathbb{R}^{n\times n}$, and $B\in\mathbb{R}^{n\times n}$ represent the joint flexibility, the actuator inertia, and the natural damping term, respectively. $u\in\mathbb{R}^n$ is the control vector used to represent the motor torque, and $T_{\mathrm{E}}\in\mathbb{R}^n$ representing an additive bounded torque disturbance. The flexible joint robot dynamics has the following properties:

Property 1 The inertia matrix is symmetric and positive definite, and satisfies the following inequalities:

$$m_1\|x\|^2\leqslant x^{\mathrm{T}}M(q)x\leqslant m_2\|x\|^2,\ \ \forall x\in\mathbb{R}^n, \quad (3)$$

where m_1 and m_2 are known positive constants.

Property 2 The centripetal coriolis matrices $V(q,\dot{q})$ have the following property:

$$x^{\mathrm{T}}(\dot{M}(q)-2V(q,\dot{q}))x=0,\ \ \forall x\in\mathbb{R}^n, \quad (4)$$

Normally, the joint elasticity matrix and motor inertia matrix are bounded. Property 1 is very important in generating a positive definite function to prove stability of the closed-loop system. Property 2 will help in simplifying the controller.

As a preliminary to the control design, we define the state space variables as $x_1=q$, $x_2=\dot{q}$, $x_3=Kq_{\mathrm{f}}$, $x_4=K\dot{q}_{\mathrm{f}}$, and then, the flexible joint robot system is described as follows:

$$\begin{cases}\dot{x}_1=x_2,\\ \dot{x}_2=F_1(x_1,x_2)+G_1(x_1)x_3,\\ \dot{x}_3=x_4,\\ \dot{x}_4=F_2(x_1,x_3,x_4)+G_2u,\end{cases} \quad (5)$$

where

$$\begin{aligned}F_1(x_1,x_2)&=-M(x_1)^{-1}(V(x_1,x_2)x_2+G(x_1)\\&\quad+F(x_2)+Kx_1+T_{\mathrm{L}}),\\ F_2(x_1,x_3,x_4)&=-KJ^{-1}(BK^{-1}x_4+x_3-Kx_1+T_{\mathrm{E}}),\\ G_1(x_1)&=M(x_1)^{-1},\ \ G_2=KJ^{-1}.\end{aligned}$$

Due to large uncertainties exist in the operation of flexible joint robots, such as unmolded dynamics, parameter perturbations and load variation, the nonlinear functions F_1, F_2, G_1, G_2 are unknown. It is simple to derive that G_1 and G_2 are symmetric and positive definite matrices, since $M(q)$ is symmetric and positive definite together with K and J are diagonal positive definite. The symmetric and positive definite property of G_1 and G_2 is very important in the stability analysis of the closed-loop system in Section 4. However, if we only define the state space variables as $x_1=q$, $x_2=\dot{q}$, $x_3=q_{\mathrm{f}}$, $x_4=\dot{q}_{\mathrm{f}}$, in the case where G_1 is $M^{-1}K$, we cannot obtain the symmetric and positive definite property of G_1, as the symmetric and positive definite of M and K cannot guarantee the symmetric and positive definite of G_1. The control objective is to design an RNN-based adaptive system (illustrated in Fig. 1) to make $x_1(t)=q(t)$ to follow a desired trajectory $x_{1\mathrm{d}}(t)=q_{\mathrm{d}}(t)$.

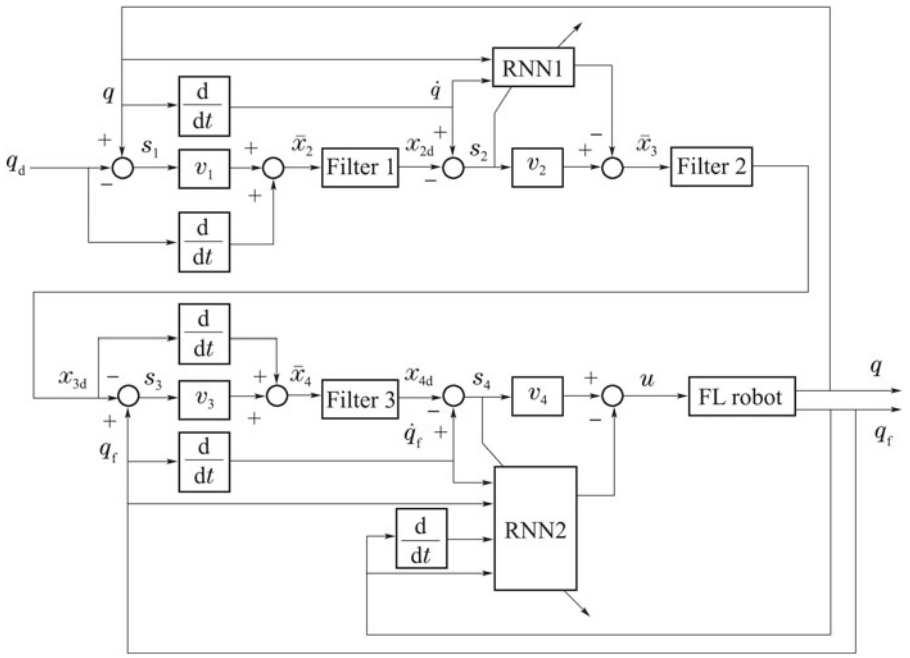


Fig. 1 RNN-based DSC scheme.

3 Recurrent neural network

A three-layer RNN, as shown in Fig. 2, which has n_1 neurons in the input layer, n_2 neurons in the hidden layer and n_3 neurons in the output layer, is adopted to implement the proposed control scheme. The signal propagation and the activation function in each layer are introduced as follows:

Layer 1 (input layer)

$$\begin{cases} \text{net}_i^1(N) = \chi_i^1(N), \\ O_i^1(N) = f_i^1(\text{net}_i^1(N)) = \dfrac{1}{1+\exp(-\text{net}_i^1(N))}, \\ i = 1, 2, \ldots, n_1, \end{cases} \quad (6)$$

where χ_i^1 represents the ith input to the node of input layer; N denotes the number of iterations; and f_i^1 is the activation function, which is a sigmoid function.

Layer 2 (hidden layer)

$$\begin{cases} \text{net}_j^2(N) = \omega_j^2 O_j^2(N-1) + \sum\limits_i \omega_{ij}^2 \chi_i^2(N), \\ O_j^2(N) = f_j^2(\text{net}_j^2(N)) = \dfrac{1}{1+\exp(-\text{net}_j^2(N))}, \\ j = 1, 2, \ldots, n_2, \end{cases} \quad (7)$$

where ω_j^2 are the weight of the self-feedback loop in the hidden layer; ω_{ij}^2 are the connective weights between the input layer and the hidden layer; and f_j^2 is the activation function, which is also a sigmoid function.

Layer 3 (output layer)

$$\begin{cases} \text{net}_k^3(N) = \sum\limits_j \omega_{jk}^3 \chi_j^3(N), \\ O_k^3(N) = f_k^3(\text{net}_k^3(N)) = \text{net}_k^3(N), \\ k = 1, 2, \ldots, n_3, \end{cases} \quad (8)$$

where ω_{jk}^3 are the connective weights between the hidden layer and the output layer; f_k^3 is the activation function, which is set to be unit; and O_k^3 the output of the RNN.

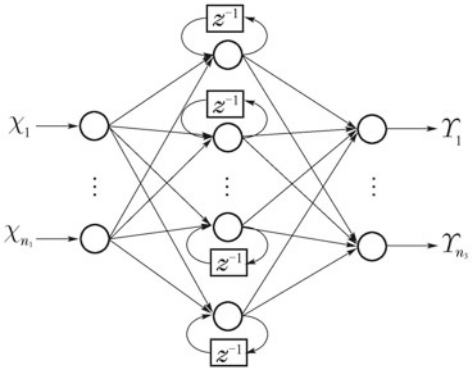


Fig. 2 Three-layer RNN structure.

Moreover, if we denote

$$\boldsymbol{\omega} = (\omega_{11}^3\omega_{21}^3\cdots\omega_{n_2 1}^3, \ldots, \omega_{1n_3}^3\omega_{2n_3}^3\cdots\omega_{n_2 n_3}^3)^{\mathrm{T}}, \quad (9)$$

$$\boldsymbol{\vartheta} = (\omega_{11}^2\omega_{21}^2\cdots\omega_{n_1 1}^2, \ldots, \omega_{1n_2}^2\omega_{2n_2}^2\cdots\omega_{n_1 n_2}^2)^{\mathrm{T}}, \quad (10)$$

$$\boldsymbol{\alpha} = [\omega_1^2 \ \ \omega_2^2 \ \cdots \ \omega_{n_2}^2]^{\mathrm{T}}, \quad (11)$$

$$\boldsymbol{\theta} = [\boldsymbol{\omega}^{\mathrm{T}} \ \ \boldsymbol{\vartheta}^{\mathrm{T}} \ \ \boldsymbol{\alpha}^{\mathrm{T}}]^{\mathrm{T}}. \quad (12)$$

The output of the RNN can be rewritten as follows:

$$O^3 = \Upsilon(\boldsymbol{\chi}, \boldsymbol{\omega}, \boldsymbol{\vartheta}, \boldsymbol{\alpha}) = \Upsilon(\boldsymbol{\chi}|\boldsymbol{\theta}), \quad (13)$$

where $\boldsymbol{\chi} = [\chi_1^1 \ \ \chi_2^1 \ \cdots \ \chi_{n_1}^1]^{\mathrm{T}}$ are the input of RNN, and $O^3 = [O_1^3 \ \ O_2^3 \ \cdots \ O_{n_3}^3]^{\mathrm{T}}$ are the output of RNN. According to the neural network approximation theory, a general function $H(\boldsymbol{\chi}) : \mathbb{R}^{n_1} \to \mathbb{R}^{n_3}$ can be written as

$$H(\boldsymbol{\chi}) = \hat{H}(\boldsymbol{\chi}|\boldsymbol{\theta}^*) + \varepsilon, \quad (14)$$

where $\hat{H}(\boldsymbol{\chi}|\boldsymbol{\theta}^*) = \Upsilon(\boldsymbol{\chi}|\boldsymbol{\theta}^*)$, ε is the functional reconstruction error; and the ideal weights $\boldsymbol{\theta}^*$ can be defined as

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}}\{\sup_{\boldsymbol{\chi}} \|H(\boldsymbol{\chi}) - \hat{H}(\boldsymbol{\chi}|\boldsymbol{\theta})\|\}. \quad (15)$$

However, the ideal weights are difficult to determine. If we denote $\hat{\boldsymbol{\theta}}$ as the estimation of $\boldsymbol{\theta}^*$, the Taylor expansion of

$\hat{H}(\boldsymbol{\chi}|\boldsymbol{\theta}^*)$ is

$$\hat{H}(\boldsymbol{\chi}|\boldsymbol{\theta}^*) = \hat{H}(\boldsymbol{\chi}|\hat{\boldsymbol{\theta}}) + \frac{\partial \hat{H}(\boldsymbol{\chi}|\hat{\boldsymbol{\theta}})}{\partial \hat{\boldsymbol{\theta}}}\tilde{\boldsymbol{\theta}} + \Delta. \quad (16)$$

where $\tilde{\boldsymbol{\theta}} = \boldsymbol{\theta}^* - \hat{\boldsymbol{\theta}}$, Δ is a vector of higher order terms; and $\frac{\partial \hat{H}(\boldsymbol{\chi}|\hat{\boldsymbol{\theta}})}{\partial \hat{\boldsymbol{\theta}}} = [\frac{\partial \hat{H}_1(\boldsymbol{\chi}|\hat{\boldsymbol{\theta}})}{\partial \hat{\boldsymbol{\theta}}} \ \frac{\partial \hat{H}_2(\boldsymbol{\chi}|\hat{\boldsymbol{\theta}})}{\partial \hat{\boldsymbol{\theta}}} \ \cdots \ \frac{\partial \hat{H}_{n_3}(\boldsymbol{\chi}|\hat{\boldsymbol{\theta}})}{\partial \hat{\boldsymbol{\theta}}}]^{\mathrm{T}}$, and $\frac{\partial \hat{H}_k(\boldsymbol{\chi}|\hat{\boldsymbol{\theta}})}{\partial \hat{\boldsymbol{\theta}}}$ $(1 \leqslant k \leqslant n_3)$ are defined as

$$\frac{\partial \hat{H}_k(\boldsymbol{\chi}|\hat{\boldsymbol{\theta}})}{\partial \hat{\boldsymbol{\theta}}} = [\frac{\partial \hat{H}_k(\boldsymbol{\chi}|\hat{\boldsymbol{\theta}})}{\partial \hat{\boldsymbol{\omega}}} \ \frac{\partial \hat{H}_k(\boldsymbol{\chi}|\hat{\boldsymbol{\theta}})}{\partial \hat{\boldsymbol{\vartheta}}} \ \frac{\partial \hat{H}_k(\boldsymbol{\chi}|\hat{\boldsymbol{\theta}})}{\partial \hat{\boldsymbol{\alpha}}}], \quad (17)$$

$$\frac{\partial \hat{H}_k(\boldsymbol{\chi}|\hat{\boldsymbol{\theta}})}{\partial \hat{\boldsymbol{\omega}}} = [\underbrace{0 \cdots 0}_{(k-1)\times n_2} \ \frac{\partial \hat{H}_k}{\partial \hat{\omega}_{1,k}} \ \cdots \ \frac{\partial \hat{H}_k}{\partial \hat{\omega}_{n_2,k}} \ \underbrace{0 \cdots 0}_{(n_3-k)\times n_2}], \quad (18)$$

$$\frac{\partial \hat{H}_k(\boldsymbol{\chi}|\hat{\boldsymbol{\theta}})}{\partial \hat{\boldsymbol{\vartheta}}} = [\underbrace{0 \cdots 0}_{(k-1)\times n_1} \ \frac{\partial \hat{H}_k}{\partial \hat{\vartheta}_{1,k}} \ \cdots \ \frac{\partial \hat{H}_k}{\partial \hat{\vartheta}_{n_1,k}} \ \underbrace{0 \cdots 0}_{(n_2-k)\times n_1}], \quad (19)$$

$$\frac{\partial \hat{H}_k(\boldsymbol{\chi}|\hat{\boldsymbol{\theta}})}{\partial \hat{\boldsymbol{\alpha}}} = [\frac{\partial \hat{H}_k}{\partial \hat{\alpha}_1} \ \cdots \ \frac{\partial \hat{H}_k}{\partial \hat{\alpha}_{n_2}}]. \quad (20)$$

Based on the above analysis, for a arbitrarily function $H(\boldsymbol{\chi})$, we can transform it into the following form:

$$H(\boldsymbol{\chi}) = \hat{H}(\boldsymbol{\chi}|\hat{\boldsymbol{\theta}}) + \frac{\partial \hat{H}(\boldsymbol{\chi}|\hat{\boldsymbol{\theta}})}{\partial \hat{\boldsymbol{\theta}}}\tilde{\boldsymbol{\theta}} + (\Delta + \varepsilon), \quad (21)$$

where $\hat{H}(\boldsymbol{\chi}|\hat{\boldsymbol{\theta}})$ can be represented by an RNN output; $[\frac{\partial \hat{H}(\boldsymbol{\chi}|\hat{\boldsymbol{\theta}})}{\partial \hat{\boldsymbol{\theta}}}]\tilde{\theta}$ is a parametric linear term respect to $\tilde{\boldsymbol{\theta}}$, and $(\Delta + \varepsilon)$ is treated as an unknown disturbance. In the following work, an adaptive law is designed for $\tilde{\boldsymbol{\theta}}$, and a H_∞ controller is embedded to attenuate the effects of $(\Delta + \varepsilon)$ on tracking performance.

4 Robust dynamic surface control

4.1 Dynamic surface control

Since the model (5) has a strict feedback form, the dynamic surface control technique is used to design the controller for flexible joint robots. Recurrent neural networks are used to adaptively learning the uncertain functions in the flexible joint robot. The design procedure is as follows.

Step 1 Design the virtual control law for x_2. The first error surface is defined as $s_1 = x_1 - x_{1\mathrm{d}}$, and then, its derivative is

$$\dot{s}_1 = x_2 - \dot{x}_{1\mathrm{d}}, \quad (22)$$

Therefore, the virtual control vector $\bar{x}_2$ is

$$\bar{x}_2 = v_1 + \dot{x}_{1\mathrm{d}}, \quad (23)$$

where v_1 is the robust control term to be designed. We pass $\bar{x}_2$ through a first-order with the time constant τ_2 to obtain $x_{2\mathrm{d}}$:

$$\tau_2 \dot{x}_{2\mathrm{d}} + x_{2\mathrm{d}} = \bar{x}_2, \ \ x_{2\mathrm{d}}(0) = \bar{x}_2(0). \quad (24)$$

Step 2 Design the virtual control law for x_3. The error surface with $x_{2\mathrm{d}}$ is defined as $s_2 = x_2 - x_{2\mathrm{d}}$, differentiating it yields

$$\begin{aligned}\dot{s}_2 &= F_1(x_1, x_2) + G_1(x_1)x_3 - \dot{x}_{2\mathrm{d}} \\ &= -G_1(x_1)V(x_1, x_2)s_2 + G_1(x_1)(x_3 + H_1(\boldsymbol{\chi}_1)). \end{aligned} \quad (25)$$

where

$$H_1(\boldsymbol{\chi}_1) = G_1^{-1}(x_1)(F_1(x_1, x_2) - \dot{x}_{2\mathrm{d}}) + V(x_1, x_2)s_2.$$

According to equation (21), $H_1(\boldsymbol{\chi}_1)$ can be rewritten as

$$H_1(\boldsymbol{\chi}_1) = \hat{H}_1(\boldsymbol{\chi}_1|\hat{\boldsymbol{\theta}}_1) + \frac{\partial \hat{H}_1(\boldsymbol{\chi}_1|\hat{\boldsymbol{\theta}}_1)}{\partial \hat{\boldsymbol{\theta}}_1}\tilde{\boldsymbol{\theta}}_1 + (\Delta_1 + \varepsilon_1). \quad (26)$$

Then, the virtual control vector $\bar{x}_3$ is chosen as

$$\bar{x}_3 = v_2 - \hat{H}_1(\boldsymbol{\chi}_1|\hat{\boldsymbol{\theta}}_1), \quad (27)$$

where v_2 is the robust control term to be designed. We pass $\bar{x}_3$ through a first-order with the time constant τ_3 to obtain

$$\tau_3 \dot{x}_{3\mathrm{d}} + x_{3\mathrm{d}} = \bar{x}_3, \ \ x_{3\mathrm{d}}(0) = \bar{x}_3(0). \quad (28)$$

Step 3 Define error surface $s_3 = x_3 - x_{3\mathrm{d}}$, and its derivative is

$$\dot{s}_3 = x_4 - \dot{x}_{3\mathrm{d}}. \quad (29)$$

The virtual control vector $\bar{x}_4$ is

$$\bar{x}_4 = v_3 + \dot{x}_{3\mathrm{d}}, \quad (30)$$

where v_3 is the robust control term to be designed. We pass $\bar{x}_4$ through a first-order with the time constant τ_4 to obtain

$$\tau_4 \dot{x}_{4\mathrm{d}} + x_{4\mathrm{d}} = \bar{x}_4, \ \ x_{4\mathrm{d}}(0) = \bar{x}_4(0). \quad (31)$$

Step 4 Design the actual control vector u. Define the fourth error surface $s_4 = x_4 - x_{4\mathrm{d}}$, and using (5), its derivative is

$$\dot{s}_4 = F_2(x_1, x_3, x_4) + G_2 u - \dot{x}_{4\mathrm{d}} = G_2(u + H_2(\boldsymbol{\chi}_2)). \quad (32)$$

where $H_2(\boldsymbol{\chi}_2) = G_2^{-1}(F_2(x_1, x_3, x_4) - \dot{x}_{4\mathrm{d}})$. According to equation (21), $H_2(\boldsymbol{\chi}_2)$ can be rewritten as

$$H_2(\boldsymbol{\chi}_2) = \hat{H}_2(\boldsymbol{\chi}_2|\hat{\boldsymbol{\theta}}_2) + \frac{\partial \hat{H}_2(\boldsymbol{\chi}_2|\hat{\boldsymbol{\theta}}_2)}{\partial \hat{\boldsymbol{\theta}}_2}\tilde{\boldsymbol{\theta}}_2 + (\Delta_2 + \varepsilon_2). \quad (33)$$

The actual control vector u is chosen as

$$u = v_4 - \hat{H}_2(\boldsymbol{\chi}_2|\hat{\boldsymbol{\theta}}_2), \quad (34)$$

where v_4 is the robust control term to be designed. So far, the dynamic surface control design procedure is finished. The overall control law should incorporate the adaptive laws for $\tilde{\boldsymbol{\theta}}_1$, $\tilde{\boldsymbol{\theta}}_2$, and the embedded H_∞ controller v_1, v_2, v_3, v_4.

4.2 H_∞ controller design

In this part, adaptive laws for $\tilde{\boldsymbol{\theta}}_1$, $\tilde{\boldsymbol{\theta}}_2$ is designed based on the Lyapunov stability theory, and an embedded H_∞ controller are designed to attenuate the effects of approximation error and filter error on the tracking performance.

First, note that

$$x_i - \bar{x}_i = x_i - x_{i\mathrm{d}} + x_{i\mathrm{d}} - \bar{x}_i = s_i + \xi_i, \ \ i = 2, 3, 4. \quad (35)$$

We denote the filter error signal as $\xi_i = x_{i\mathrm{d}} - \bar{x}_i$, $\xi_i \to 0$ if the time constant τ_i is small enough. In the standard stability analysis of dynamic surface control method, ξ_i is treated as new state variables, which complicates the matters. In this paper, we treat filter error ξ_i as external disturbance as well as approximation error $(\Delta_i + \varepsilon_i)$. Thus, a robust controller is needed to surpass them. Based on the preceding analysis, we have

$$\begin{cases} \dot{s}_1 = v_1 + s_2 + \omega_1, \\ \dot{s}_2 = -G_1 V s_2 + G_1(v_2 + s_3 + \dfrac{\partial \hat{H}_1(\boldsymbol{\chi}_1|\hat{\boldsymbol{\theta}}_1)}{\partial \hat{\boldsymbol{\theta}}_1}\tilde{\boldsymbol{\theta}}_1 + \omega_2), \\ \dot{s}_3 = v_3 + s_4 + \omega_3, \\ \dot{s}_4 = G_2(v_4 + \dfrac{\partial \hat{H}_2(\boldsymbol{\chi}_2|\hat{\boldsymbol{\theta}}_2)}{\partial \hat{\boldsymbol{\theta}}_2}\tilde{\boldsymbol{\theta}}_2 + \omega_4), \end{cases} \quad (36)$$

where

$$\omega_1 = \xi_2,\ \omega_2 = \Delta_1 + \varepsilon_1 + \xi_3,\ \omega_3 = \xi_4,\ \omega_4 = \Delta_2 + \varepsilon_2.$$

For ease of notation, we define

$$\begin{aligned}\boldsymbol{\omega} &= [\omega_1^{\mathrm{T}}\ \ \omega_2^{\mathrm{T}}\ \ \omega_3^{\mathrm{T}}\ \ \omega_4^{\mathrm{T}}]^{\mathrm{T}},\\ \boldsymbol{v} &= [v_1^{\mathrm{T}}\ \ v_2^{\mathrm{T}}\ \ v_3^{\mathrm{T}}\ \ v_4^{\mathrm{T}}]^{\mathrm{T}},\\ \boldsymbol{s} &= [s_1^{\mathrm{T}}\ \ s_2^{\mathrm{T}}\ \ s_3^{\mathrm{T}}\ \ s_4^{\mathrm{T}}]^{\mathrm{T}},\end{aligned}$$

and then, the main results are summarized as follows:

Theorem 1 Consider the system represented by (36), the control system, illustrated in Fig. 1, is designed as (23), (27), (30) and (34), in which the adaptive laws and the embedded H_∞ controller are designed as

$$\dot{\hat{\boldsymbol{\theta}}}_1 = \Gamma_1[\frac{\partial \hat{H}_1(\boldsymbol{\chi}_1|\hat{\boldsymbol{\theta}}_1)}{\partial \hat{\boldsymbol{\theta}}_1}]^{\mathrm{T}} s_2, \tag{37}$$

$$\dot{\hat{\boldsymbol{\theta}}}_2 = \Gamma_2[\frac{\partial \hat{H}_2(\boldsymbol{\chi}_2|\hat{\boldsymbol{\theta}}_2)}{\partial \hat{\boldsymbol{\theta}}_2}]^{\mathrm{T}} s_4, \tag{38}$$

$$\boldsymbol{v} = -(\frac{1}{2} + \frac{1}{2\gamma^2} + \Lambda)\boldsymbol{s}. \tag{39}$$

Then, the overall control scheme guarantees the following H_∞ tracking performance: $\forall T > 0,\ \beta \in \mathbb{R}$,

$$\int_0^T \boldsymbol{s}^{\mathrm{T}}(t)\boldsymbol{s}(t)\mathrm{d}t \leqslant \gamma^2 \int_0^T \boldsymbol{\omega}^{\mathrm{T}}(t)\boldsymbol{\omega}(t)\mathrm{d}t + \beta, \tag{40}$$

where Γ_1, Γ_2 are symmetric and positive definite matrices, γ is a prescribed attenuation constant; and matrix Λ is defined as

$$\Lambda = \begin{bmatrix} 0 & & & \\ 1 & 0 & & \\ & \ddots & \ddots & \\ & & 1 & 0 \end{bmatrix}. \tag{41}$$

Proof Since G_1 and G_2 are symmetric and positive definite matrices, we consider the following Lyapunov function:

$$\begin{aligned}V = &\frac{1}{2}s_1^{\mathrm{T}}s_1 + \frac{1}{2}s_2^{\mathrm{T}}G_1^{-1}s_2 + \frac{1}{2}s_3^{\mathrm{T}}s_3\\ &+\frac{1}{2}s_4^{\mathrm{T}}G_2^{-1}s_4 + \frac{1}{2}\sum_{i=1}^{2}\tilde{\boldsymbol{\theta}}_i^{\mathrm{T}}\Gamma_i^{-1}\tilde{\boldsymbol{\theta}}_i.\end{aligned} \tag{42}$$

Along the trajectories of (36), the time derivative of V is given as

$$\begin{aligned}\dot{V} =& s_1^{\mathrm{T}}\dot{s}_1 + s_2^{\mathrm{T}}G_1^{-1}\dot{s}_2 + \frac{1}{2}s_2^{\mathrm{T}}\dot{G}_1^{-1}s_2\\ &+s_3^{\mathrm{T}}\dot{s}_3 + s_4^{\mathrm{T}}G_2^{-1}\dot{s}_4 + \sum_{i=1}^{2}\dot{\tilde{\boldsymbol{\theta}}}_i^{\mathrm{T}}\Gamma_i^{-1}\tilde{\boldsymbol{\theta}}_i\\ =& s_1^{\mathrm{T}}(v_1 + s_2 + \omega_1) + s_2^{\mathrm{T}}(v_2 + s_3 + \frac{\partial \hat{H}_1(\boldsymbol{\chi}_1|\hat{\boldsymbol{\theta}}_1)}{\partial \hat{\boldsymbol{\theta}}_1}\tilde{\boldsymbol{\theta}}_1\\ &+\omega_2) + \frac{1}{2}s_2^{\mathrm{T}}(\dot{G}_1^{-1} - 2V)s_2 + s_3^{\mathrm{T}}(v_3 + s_4 + \omega_3)\\ &+s_4^{\mathrm{T}}(v_4 + \frac{\partial \hat{H}_2(\boldsymbol{\chi}_2|\hat{\boldsymbol{\theta}}_2)}{\partial \hat{\boldsymbol{\theta}}_2}\tilde{\boldsymbol{\theta}}_2 + \omega_4) + \sum_{i=1}^{2}\dot{\tilde{\boldsymbol{\theta}}}_i^{\mathrm{T}}\Gamma_i^{-1}\tilde{\boldsymbol{\theta}}_i\\ =& \boldsymbol{s}^{\mathrm{T}}(\boldsymbol{v}+\boldsymbol{\omega}) + \boldsymbol{s}^{\mathrm{T}}\Lambda\boldsymbol{s} + (\dot{\tilde{\boldsymbol{\theta}}}_1^{\mathrm{T}}\Gamma_1^{-1} + s_2^{\mathrm{T}}\frac{\partial \hat{H}_1(\boldsymbol{\chi}_1|\hat{\boldsymbol{\theta}}_1)}{\partial \hat{\boldsymbol{\theta}}_1})\tilde{\boldsymbol{\theta}}_1\\ &+(\dot{\tilde{\boldsymbol{\theta}}}_2^{\mathrm{T}}\Gamma_2^{-1} + s_4^{\mathrm{T}}\frac{\partial \hat{H}_2(\boldsymbol{\chi}_2|\hat{\boldsymbol{\theta}}_2)}{\partial \hat{\boldsymbol{\theta}}_2})\tilde{\boldsymbol{\theta}}_2.\end{aligned} \tag{43}$$

Using (37) and (38), we have

$$\dot{V} \leqslant \boldsymbol{s}^{\mathrm{T}}\boldsymbol{v} + \boldsymbol{s}^{\mathrm{T}}\Lambda\boldsymbol{s} + \boldsymbol{s}^{\mathrm{T}}\boldsymbol{\omega}. \tag{44}$$

From Young's inequality: $ab \leqslant \frac{1}{2\gamma^2}a^2 + \frac{\gamma^2}{2}b^2,\ a, b \in \mathbb{R}$, the following inequalities hold:

$$\boldsymbol{s}^{\mathrm{T}}\boldsymbol{\omega} \leqslant \frac{1}{2\gamma^2}\boldsymbol{s}^{\mathrm{T}}\boldsymbol{s} + \frac{\gamma^2}{2}\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\omega}. \tag{45}$$

Then, substituting (39) and (45) into (44) yields

$$\begin{aligned}\dot{V} &\leqslant \boldsymbol{s}^{\mathrm{T}}\boldsymbol{v} + \boldsymbol{s}^{\mathrm{T}}\Lambda\boldsymbol{s} + \frac{1}{2\gamma^2}\boldsymbol{s}^{\mathrm{T}}\boldsymbol{s} + \frac{\gamma^2}{2}\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\omega}\\ &= \boldsymbol{s}^{\mathrm{T}}(\boldsymbol{v} + \Lambda\boldsymbol{s} + \frac{1}{2\gamma^2}\boldsymbol{s}) + \frac{\gamma^2}{2}\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\omega}\\ &= -\frac{1}{2}\boldsymbol{s}^{\mathrm{T}}\boldsymbol{s} + \frac{\gamma^2}{2}\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\omega}.\end{aligned} \tag{46}$$

Integrating the above inequality from $t = 0$ to $t = T$ yields

$$\begin{aligned}V(T) - V(0) \leqslant &-\frac{1}{2}\int_0^{\mathrm{T}}\boldsymbol{s}^{\mathrm{T}}(t)\boldsymbol{s}(t)\mathrm{d}t\\ &+\frac{\gamma^2}{2}\int_0^{\mathrm{T}}\boldsymbol{\omega}^{\mathrm{T}}(t)\boldsymbol{\omega}(t)\mathrm{d}t.\end{aligned} \tag{47}$$

Since $V(T) \geqslant 0$, (47) implies

$$\int_0^{\mathrm{T}}\boldsymbol{s}^{\mathrm{T}}(t)\boldsymbol{s}(t)\mathrm{d}t \leqslant 2V(0) + \gamma^2\int_0^{\mathrm{T}}\boldsymbol{\omega}^{\mathrm{T}}(t)\boldsymbol{\omega}(t)\mathrm{d}t. \tag{48}$$

Thus, the theorem is proved.

Remark 1 If $\boldsymbol{\omega}(t)$ is squared integrable, that is, $\int_0^{\mathrm{T}}\boldsymbol{\omega}^{\mathrm{T}}(t)\boldsymbol{\omega}(t)\mathrm{d}t < \infty$, and then, $\lim\limits_{t\to\infty}\|\boldsymbol{s}(t)\| = 0$. It is can be easily obtained by using Barbalat's lemma. In addition, if $V(0) = 0$, the H_∞ tracking performance (40) can be rewritten as

$$\sup_{\boldsymbol{\omega}\in L_2[0,T]}\frac{\|\boldsymbol{s}\|}{\|\boldsymbol{\omega}\|} \leqslant \gamma, \tag{49}$$

which means the L_2-gain from $\boldsymbol{\omega}$ to $\boldsymbol{s}$ must be equal to or less than a level γ.

5 Simulation results

The single link flexible joint robot (in Fig. 3) is simulated to examine the effectiveness of the proposed control scheme. The model can be written as

$$I\ddot{q} + Mgl\sin q + B_1\dot{q} + Kq = Kq_{\mathrm{f}}, \tag{50}$$

$$I\ddot{q}_{\mathrm{f}} + B_2\dot{q}_{\mathrm{f}} + K(q_{\mathrm{f}} - q) = u, \tag{51}$$

where $I = 0.1\,\mathrm{kg}\cdot\mathrm{m}^2$, $J = 1\,\mathrm{kg}\cdot\mathrm{m}^2$, $M = 1\,\mathrm{kg}$, $L = 0.1$ m; g$= 9.8\,\mathrm{m}\cdot\mathrm{s}^{-2}$, $K = 100\,\mathrm{Nm}\cdot\mathrm{rad}^{-1}$, $B_1 = B_2 = 1\,\mathrm{Nm}\cdot\mathrm{rad}^{-1}$. The desired trajectory is $q_{\mathrm{d}}(t) = \sin(2t)$. Two recurrent neural networks are used to adaptively learning the uncertain functions $H_1(\boldsymbol{\chi}_1)$ and $H_2(\boldsymbol{\chi}_2)$. The input of the RNNs are $\boldsymbol{\chi}_1 = [q\ \ \dot{q}\ \ q_{\mathrm{d}}]^{\mathrm{T}}$ and $\boldsymbol{\chi}_2 = [q\ \ \dot{q}\ \ q_{\mathrm{f}}\ \ \dot{q}_{\mathrm{f}}\ \ q_{\mathrm{d}}]^{\mathrm{T}}$, respectively, and the number of neurons in the hidden layer both are 10. All the weights are initialize randomly, other parameters used in the control scheme are: $\tau_2 = \tau_3 = \tau_4 = 0.05$, $\Gamma_1 = 1000$, $\Gamma_2 = 1000$.

To evaluate the effect of different values of parameter γ on the tracking performance, we define the mean square error as

$$\mathrm{MSE} = \frac{1}{K}\sum_{k=1}^{K}e^2(k), \tag{52}$$

where $e(k) = q(k) - q_{\mathrm{d}}(k)$ is the tracking error in the kth sample time, K is the total sample number, which set $K = 100000$ in this simulation. Two cases $\gamma = 0.25$

and $\gamma = 0.2$ are simulated to examine the effectiveness of the proposed control scheme. The simulation results for $\gamma = 0.25$ is illustrated in Figs. 4–6, and Figs. 7–9 for $\gamma = 0.2$. The mean square error for $\gamma = 0.25$ is 0.0003, while 0.0001 for the $\gamma = 0.2$ case. From the simulation results, it can be concluded that the design of the controller in this paper has good tracking performance, tracking error may be limited to less than 5%, and has a fast response. In addition, the mean square error for $\gamma = 0.2$ is less than the mean square error for $\gamma = 0.25$, which indicates that it has better tracking performance when $\gamma = 0.2$. This is consistent with the physical meaning of parameter γ. We can explain the results by high gain feedback, since from (39), we can see that smaller γ gives higher feedback gain, which results in smaller tracking error.

For a comparison of performance, the simulation results of composite control method, proposed in [5], are given in Figs. 10 and 11. The smallest MSE of several trails is 0.0032, which indicate that the proposed RNN-based controller outperforms the composite control method.

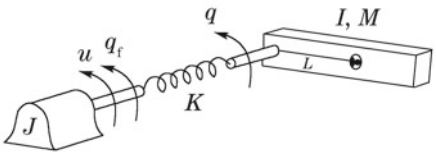


Fig. 3 Model of single link flexible joint robot.

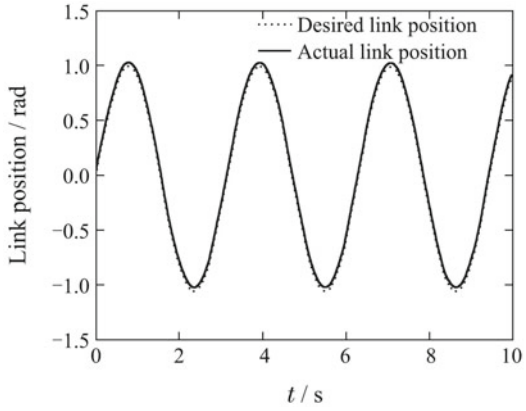


Fig. 4 Actual $q(t)$ and desired $q_{\mathrm{d}}(t)$ when $\gamma = 0.25$.

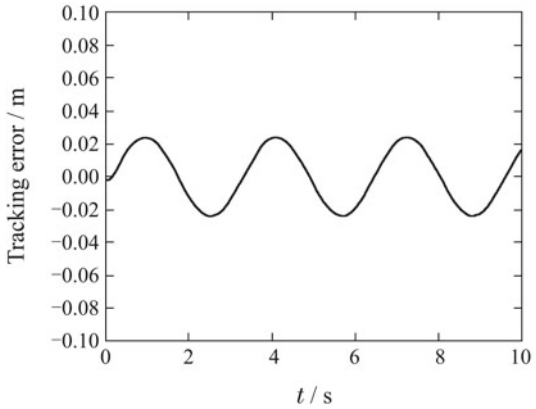


Fig. 5 Tracking error $e(t)$ when $\gamma = 0.25$ (MSE $= 0.0003$).

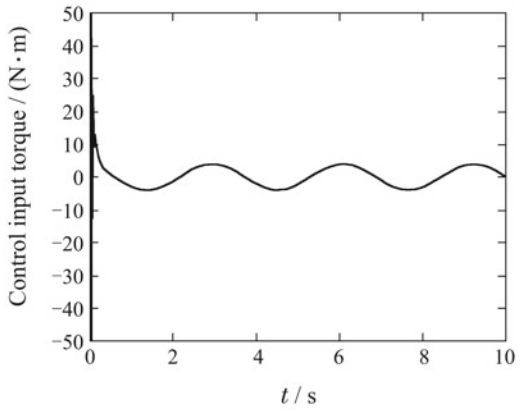


Fig. 6 Motor torque control input $u(t)$ when $\gamma = 0.25$.

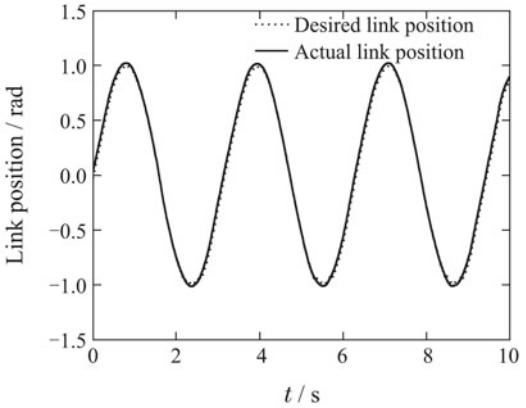


Fig. 7 Actual $q(t)$ and desired $q_{\mathrm{d}}(t)$ when $\gamma = 0.2$.

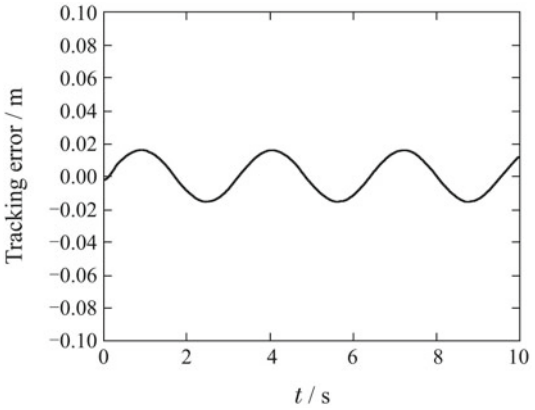


Fig. 8 Tracking error $e(t)$ when $\gamma = 0.2$ (MSE $= 0.0001$).

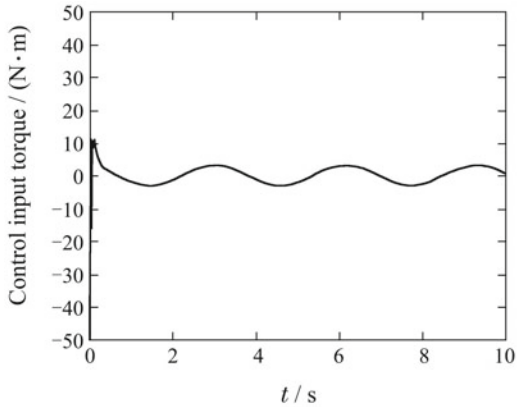


Fig. 9 Motor torque control input $u(t)$ when $\gamma = 0.2$.

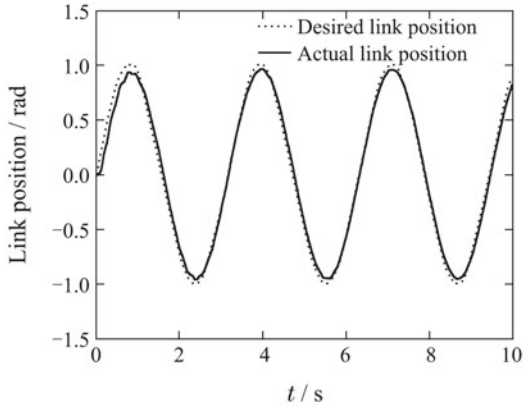


Fig. 10 Tracking performance of composite controller.

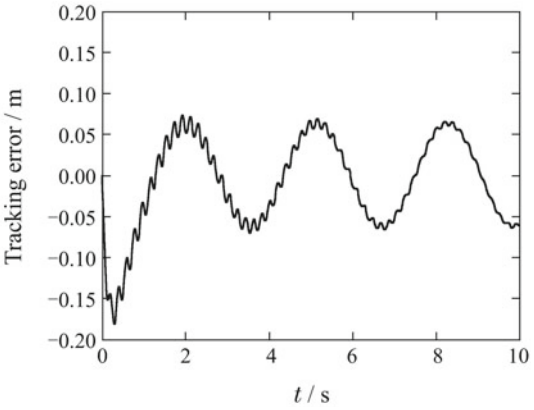


Fig. 11 Tracking error of composite controller ($\mathrm{MSE} = 0.0032$).

6 Conclusions

In this paper, a robust control system using RNNs approximators and adaptive DSC technique for flexible joint robots with model uncertainties has been developed. Two RNNs are adopted to adaptively learning the uncertain functions in the flexible joint robot. Then, the effects of the approximation error and filter error are attenuated to a prescribed level by the embedded H_∞ controller. Though Lyapunov stability analysis and H_∞ design method, the adaptation laws for all weights of RNNs and the embedded H_∞ controller have been derived, which guarantees the desired H_∞ tracking performance can be achieved. Finally, a simulation study for a single link flexible joint robot manipulator shows that the proposed control scheme has good tracking performance.

References

[1] K. Khorasani. Nonlinear feedback control of flexible joint manipulators: a single link case study. *IEEE Transactions on Automatic Control*, 1990, 35(10): 1145 – 1149.

[2] M. W. Spong, M. V. Vidyasager. *Robot Dynamics and Control*. Singapore: John Wiley, 1989: 281 – 295.

[3] F. Ghorbel, J. Hung, M. W. Spong. Adaptive control of flexible- joint manipulators. *IEEE Control System Magazine*, 1989, 9(7): 9 – 13.

[4] M. W. Spong, K. Khorasani, P. V. Koktovic. An integral manifold approach to the feedback control of flexible-joint robots. *IEEE Robotics and Automation*, 1987, 3(4): 291 – 300.

[5] F. Ghorbel, M. W. Spong. Adaptive integral manifold control of flexible joint robot manipulators. *Proceedings of IEEE Internatinal Conference on Robotics and Automation*. New York: IEEE, 1992: 707 – 714.

[6] D. G. Taylor. Composite control of direct-drive robots. *IEEE Proceedings of the 28th Conference on Decision and Control*. New York: IEEE, 1989: 1670 – 1675.

[7] A. A. Abouelsoud. Robust regulator for flexible-joint robots using integrator backstepping. *Journal of Intelligent and Robotic Systems*, 1998, 22(1): 23 – 38.

[8] H. O. Jong, J. S. Lee. Control of flexible joint robot system by backstepping design approach. *Proceedings of the IEEE International Conference on Robotics and Automation*. New York: IEEE, 1997: 3435 – 3440.

[9] M. M. Bridges, D. M. Dawson, C. T. Abdallah. Control of rigid-link flexible-joint robots: a survey of backstepping approaches. *Journal Robotic Systems*, 1995, 12(3): 199 – 216.

[10] C. A. Lightcap, S. A. Banks. An extended Kalman filter for real-time estimation and control of a rigid-link flexible-joint manipulator. *IEEE Transactions on Control Systems Technology*, 2010, 18(1): 91 – 103.

[11] C. M. Kwan, F. L. Lewis, Y. H. Kim. Robust neural network control of rigid link flexible joint robots. *Asian Journal of Control*, 1999, 1(3): 188 – 197.

[12] C. M. Kwan, F. L. Lewis. Robust backstepping control of nonlinear systems using neural networks. *IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans*, 2000, 30(6): 753 – 766.

[13] T. Zhang, S. S. Ge, C. Hang. Adaptive neural network control for strict-feedback nonlinear systems using backstepping design. *Automatica*, 2000, 36(12): 1835 – 1846.

[14] S. S. Ge, C. Wang. Direct adaptive NN control of a class of nonlinear systems. *IEEE Transactions on Neural Networks*, 2002, 13(1): 214 – 221.

[15] Y. Li, S. Qiang, X. Zhuang, et al. Robust and adaptive backstepping control for nonlinear systems using RBF neural networks. *IEEE Transactions on Neural Networks*, 2004, 15(3): 693 – 701.

[16] D. Swaroop, J. C. Gerdes, P. P. Yip, et al. Dynamic surface control of nonlinear systems. *Proceedings of the American Control Conference*. New York: IEEE, 1997: 3028 – 3034.

[17] D. Swaroop, J. K. Hedrick, P. P. Yip, et al. Dynamic surface control for a class of nonlinear systems. *IEEE Transactions on Automatic Control*, 2000, 45(10): 1893 – 1899.

[18] D. Wang, J. Huang. Neural network-based adaptive dynamic surface control for a class of uncertain nonlinear systems in strict-feedback form. *IEEE Transactions on Neural Networks*, 2005, 16(1): 195 – 202.

[19] W. Chen. Adaptive backstepping dynamic surface control for systems with periodic disturbances using neural networks. *IET Control Theory and Applications*, 2009, 3(10): 1383 – 1394.

[20] S. J. Yoo, J. B. Park, Y. H. Choi. Adaptive dynamic surface control of flexible-joint robots using self-recurrent wavelet neural networks. *IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics*, 2006, 36(6): 1342 – 1355.

[21] S. J. Yoo, J. B. Park, Y. H. Choi. Adaptive output feedback control of flexible-joint robots using neural networks: dynamic surface design approach. *IEEE Transactions on Neural Networks*, 2008, 19(10): 1712 – 1726.

[22] S. S. Ge, C. Wang. Adaptive neural control of uncertain MIMO nonlinear systems. *IEEE Transactions on Neural Networks*, 2004, 15(3): 674 – 692.

[23] C. Lin, C. F. Hsu. Recurrent neural network based adaptive backstepping control for induction servo motors. *IEEE Transactions on Industrial Elctronics*, 2005, 52(6): 1677 – 1684.

[24] C. Lin, W. D. Chou, F. Lin. Adaptive hybrid control using a recurrent neural network for a linear synchronous motor servo-drive system. *IEE Proceedings – Control Theory and Applications*, 2001, 148(2): 156 – 168.

[25] F. Lin, R. Wai, W. Chou, et al. Adaptive backstepping control using recurrent neural network for linear induction motor drive. *IEEE Transactions on Industrial Electronics*, 2002, 49(1): 134 – 146.

[26] C. Ku, K. Y. Lee. Diagonal recurrent neural networks for dynamic systems control. *IEEE Transactions on Neural Networks*, 1995, 6(1): 144 – 156.

[27] C. Lin, C. Chen. CMAC-based supervisory control for nonlinear chaotic systems. *Chaos, Solitons and Fractals*, 2008, 35(1): 40 – 58.

[28] C. Lin, Y. Peng, C. F. Hsu. Robust cerebellar model articulation controller design for unknown nonlinear systems. *IEEE Transactions on Circuits and Systems – II: Express Briefs*, 2004, 51(7): 354 – 358.

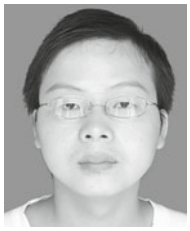
[29] F. Lin, T. S. Lee, C. Lin. Robust controller design with recurrent neural network for linear synchronous motor drive. *IEEE Transactions on Industrial Electronics*, 2003, 50(3): 456 – 470.

[30] C. Chen, C. Lin, T. Chen. Intelligent adaptive control for MIMO uncertain nonlinear systems. *Expert Systems with Applications*, 2008, 35(3): 865 – 877.

[31] C. F. Hsu, C. Lin, T. T. Lee. Wavelet adaptive backstepping control for a class of nonlinear systems. *IEEE Transactions on Neural Networks*, 2006, 17(5): 1175 – 1183.

[32] M. C. Hwang, X. Hu, Y. Shrivastava. Adaptive neural network tracking controller for electrically driven manipulators. *IEE Proceedings – Control Theory and Applications*, 1998, 145(6): 594 – 602.

[33] S. J. Yoo, J. B. Park. Practical robust control for flexible joint robot manipulators. *IEEE International Conference on Robotics and Automation*. New York: IEEE, 2008: 3377 – 3382.



Zhiqiang MIAO received his B.S. and M.S. degrees in Electrical and Information Engineering from Hunan University, Changsha, China, in 2010 and 2012, respectively, where he is currently working toward the Ph.D. degree with the College of Electrical and Information Engineering. His research interests include intelligent control theory and application and robot control. E-mail: miaozhiqiang@hnu.edu.cn.

Yaonan WANG received his B.S. degree in Computer Engineering from East China Science and Technology University (ECSTU), Fuzhou, China, in 1981, and M.S. and Ph.D. degrees in Electrical Engineering from Hunan University, Changsha, China, in 1990 and 1994, respectively. From 1994 to 1995, he was a postdoctoral research fellow with the National University of Defence Technology. From 1981 to 1994, he worked with ECSTU. From 1998 to 2000, he was a senior Humboldt fellow in Germany, and from 2001 to 2004, he was a visiting professor with the University of Bremen, Bremen, Germany. He has been a professor at Hunan University since 1995. His research interests include intelligent control and information processing, robot control, industrial process control, and image processing. E-mail: yaonan@hnu.edu.cn.