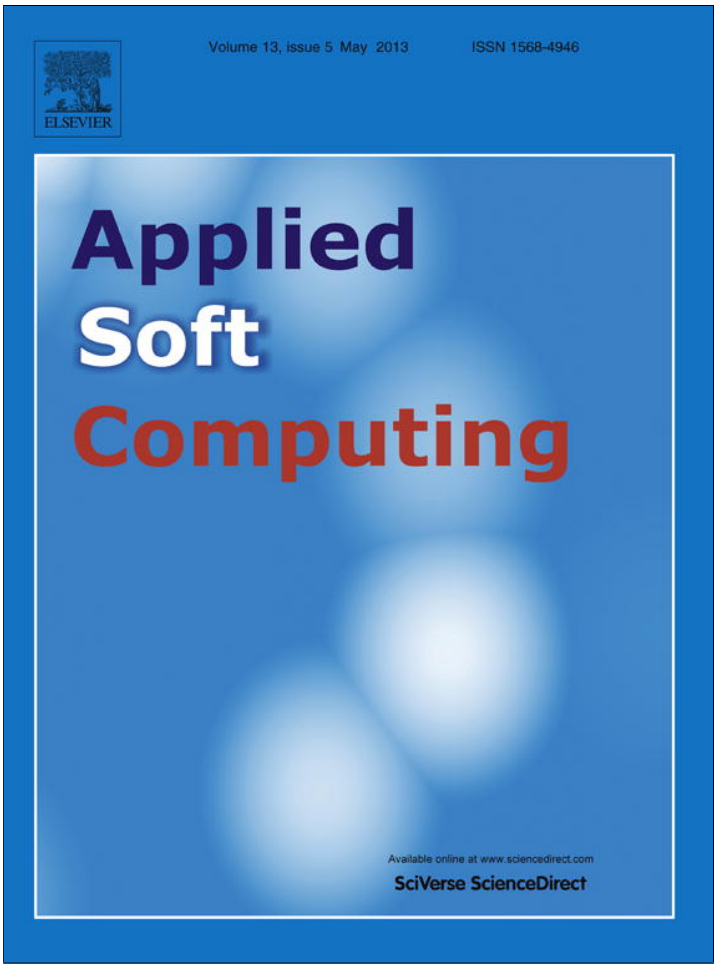
Volume 13, issue 5 May 2013
ISSN 1568-4946
ELSEVIER
Applied Soft Computing
Available online at www.sciencedirect.com
SciVerse ScienceDirect

Applied Soft Computing 13 (2013) 2569–2576

Contents lists available at SciVerse ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Adaptive PI Hermite neural control for MIMO uncertain nonlinear systems

Chun-Fei Hsu

Department of Electrical Engineering, Tamkang University, No. 151, Yingzhuan Rd., Tamsui Dist., New Taipei City 25137, Taiwan, ROC

ARTICLE INFO

Article history:
Received 23 November 2011
Received in revised form 14 July 2012
Accepted 22 November 2012
Available online 8 December 2012

Keywords:
Hermite neural network
Adaptive control
Neural control
Pendulum
Robotic manipulator

ABSTRACT

This paper presents an adaptive PI Hermite neural control (APIHNC) system for multi-input multi-output (MIMO) uncertain nonlinear systems. The proposed APIHNC system is composed of a neural controller and a robust compensator. The neural controller uses a three-layer Hermite neural network (HNN) to online mimic an ideal controller and the robust compensator is designed to eliminate the effect of the approximation error introduced by the neural controller upon the system stability in the Lyapunov sense. Moreover, a proportional–integral learning algorithm is derived to speed up the convergence of the tracking error. Finally, the proposed APIHNC system is applied to an inverted double pendulums and a two-link robotic manipulator. Simulation results verify that the proposed APIHNC system can achieve high-precision tracking performance. It should be emphasized that the proposed APIHNC system is clearly and easily used for real-time applications.

1. Introduction

An ideal controller requires an exact model of physical systems to achieve favorable control performance [1]. There is a trade-off between control performance and model accuracy for an ideal controller design. But, it is difficult for a designer to obtain an exact model of physical systems. To attack this problem, several adaptive neural control techniques have proposed to solve the unknown nonlinear control problems without significant prior knowledge of system dynamics [2–8]. The characteristics of adaptive neural controllers are robust and capable of online learning. In the last few decades, a number of works are found to focus on the multi-input multi-output (MIMO) nonlinear control problem based on adaptive neural control [9–11]. It is one of the most challenging tasks for many control engineers, especially when the nonlinear system is required to maneuver very quickly under external disturbances.

The hidden neurons of neural networks in [2–11] use the same activation functions such as sigmoid or radial basis functions. Much research has been done on the applications of orthogonal neural networks which use different activation functions for different hidden neuron [12–15]. It has been reported in various fields such as biomedical engineering, signal processing, image processing, pattern classification and control algorithm because of their distinguished approximated ability. In [12–14], a one-hidden-layer neural network was proposed in which each hidden neuron employs a different orthogonal Hermite polynomial basis function for its activation function. The learning ability of HNN is effective with high convergence precision and fast convergence time. In [15], a Hermite polynomial-based recurrent neural network is proposed to control a thrust active magnetic bearing system. Though the control performances are acceptable, the parameter adaptation law makes the convergence of tracking error slow.

Further, since the number of hidden neurons is finite in real-time approaches, the approximation error is inevitable when it is used to approximate an ideal controller or a system uncertainty. To ensure the stability of closed-loop control system, a compensator should be designed to dispel the influence of external disturbances and approximation errors. The most frequently used compensator is in a sliding-mode type form which requires the bound of the approximation error [16,17]. If the bound of approximation error is chosen too small, it cannot guarantee the system stability. Otherwise, if the bound of approximation error is chosen large to avoid instability, it will result in substantial chattering in the control effort. To reduce the chattering phenomenon, a saturation compensator is proposed in [18]; however, a trade-off problem between chattering phenomenon and control accuracy arises.

To attack this problem, Lin et al. [19] uses a fuzzy system to estimate the approximation error; however, the fuzzy rules should be pre-constructed by trial-and-error tuning procedure. In [20,21], a fuzzy compensator which possesses the advantages of simple control framework, stable tracking control performance and robust to uncertainties is proposed. But, the adaptive law will make the fuzzy compensator go to infinity. In [22–24], a robust control theory is used to attenuate the effects of approximation error. A better control performance can be achieved as a specified attenuation level is chosen smaller but it leads to a large control signal. In [25], an adaptive PI compensator is designed to cope with the bounded large-and-fast disturbances with unknown bound; however, an

E-mail address: fei@ee.tku.edu.tw

1568-4946/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.asoc.2012.11.029

indefinite steady-state error is caused depending on the selection of the boundary layer.

In this paper, a Hermite neural network (HNN) which adopts a Hermite polynomials basis function as the activation function in each hidden neuron is studied. The training algorithm typically converges in a smaller number of iterations for HNN than that for a conventional neural network. As a result, this paper proposes an adaptive PI Hermite neural control (APIHNC) system for MIMO uncertain nonlinear systems. From the Lyapunov stability analysis, the proposed APIHNC system can guarantee that the control tracking error converges to zero even though the approximation error exists. Finally, the proposed APIHNC system is applied to an inverted double pendulums and a two-link robotic manipulator. The simulation results show that high-quality tracking performance could be achieved using the proposed APIHNC scheme after the controller parameters learning. The remainder of this paper is organized as follows. Section 2 introduces the tracking control problem of MIMO nonlinear systems. Section 3 presents the HNN structure. Section 4 develops the APIHNC system and proves its stability in the Lyapunov sense. Section 5 provides simulation results and Section 6 offers a concluding remark.

2. Problem statement

Consider a second-order MIMO nonlinear systems described by the following form

$$\ddot{\mathbf{x}} = \mathbf{f}(\underline{\mathbf{x}}, t) + \mathbf{G}(\underline{\mathbf{x}}, t)\mathbf{u} \tag{1}$$

where $\mathbf{x} = [x_1, x_2, \ldots, x_m]^T$ is the system state vector, $\underline{\mathbf{x}} = [\mathbf{x}^T, \dot{\mathbf{x}}^T]^T$ and it is assumed to be available for measurement, $\mathbf{f}(\underline{\mathbf{x}}, t)$ and $\mathbf{G}(\underline{\mathbf{x}}, t)$ are the uncertain nonlinear system dynamics and $\mathbf{u} = [u_1, u_2, \ldots, u_m]^T$ is the control input vector. When neglecting the modeling uncertainties, the nominal model of MIMO nonlinear systems can be represented as

$$\ddot{\mathbf{x}} = \mathbf{f}_n(\underline{\mathbf{x}}, t) + \mathbf{G}_n(\underline{\mathbf{x}}, t)\mathbf{u} \tag{2}$$

where $\mathbf{f}_n(\underline{\mathbf{x}}, t)$ and $\mathbf{G}_n(\underline{\mathbf{x}}, t)$ are the nominal parts of $\mathbf{f}(\underline{\mathbf{x}}, t)$ and $\mathbf{G}(\underline{\mathbf{x}}, t)$, respectively. When the modeling uncertainties and external disturbance exist, the uncertain nonlinear system (1) can be formulated as

$$\begin{aligned}\ddot{\mathbf{x}} &= [\mathbf{f}_n(\underline{\mathbf{x}}, t) + \Delta\mathbf{f}(\underline{\mathbf{x}}, t)] + [\mathbf{G}_n(\underline{\mathbf{x}}, t) + \Delta\mathbf{G}(\underline{\mathbf{x}}, t)]\mathbf{u} + \mathbf{d}(\underline{\mathbf{x}}, t) \\ &= \mathbf{f}_n(\underline{\mathbf{x}}, t) + \mathbf{G}_n(\underline{\mathbf{x}}, t)\mathbf{u} + \mathbf{z}(\underline{\mathbf{x}}, t)\end{aligned} \tag{3}$$

where $\Delta\mathbf{f}(\underline{\mathbf{x}}, t)$ and $\Delta\mathbf{G}(\underline{\mathbf{x}}, t)$ are the unknown system uncertainties of $\mathbf{f}(\underline{\mathbf{x}}, t)$ and $\mathbf{G}(\underline{\mathbf{x}}, t)$, respectively, $\mathbf{d}(\underline{\mathbf{x}}, t)$ is external disturbance and $\mathbf{z}(\underline{\mathbf{x}}, t) = \Delta\mathbf{f}(\underline{\mathbf{x}}, t) + \Delta\mathbf{G}(\underline{\mathbf{x}}, t)\mathbf{u}(t) + \mathbf{d}(\underline{\mathbf{x}}, t)$ is the lump of system uncertainty. The control objective is to find a control law so that the system state vector $\mathbf{x}$ can track a reference trajectory command $\mathbf{x}_c$ closely. Then, define a tracking error vector as

$$\underline{\mathbf{e}} = [\mathbf{e}^T, \dot{\mathbf{e}}^T]^T \tag{4}$$

where $\mathbf{e} = \mathbf{x}_c - \mathbf{x}$. Assume that the lump of system uncertainty is known, there exists an ideal controller as [1]

$$\mathbf{u}^* = [u_1^*, u_2^*, \ldots, u_m^*]^T = \mathbf{G}_n^{-1}(\underline{\mathbf{x}}, t)[-\mathbf{f}_n(\underline{\mathbf{x}}, t) + \ddot{\mathbf{x}}_c + \underline{\mathbf{K}}^T\underline{\mathbf{e}} - z(\underline{\mathbf{x}}, t)] \tag{5}$$

where u_k^* is the kth ideal controller of control system and $\underline{\mathbf{K}} = [\mathbf{K}_1, \mathbf{K}_2]^T$ is the feedback gain matrix which contains real numbers. Applying the ideal controller (5) into (3), it is obtained that

$$\dot{\underline{\mathbf{e}}} = \mathbf{\Lambda}\underline{\mathbf{e}} \tag{6}$$

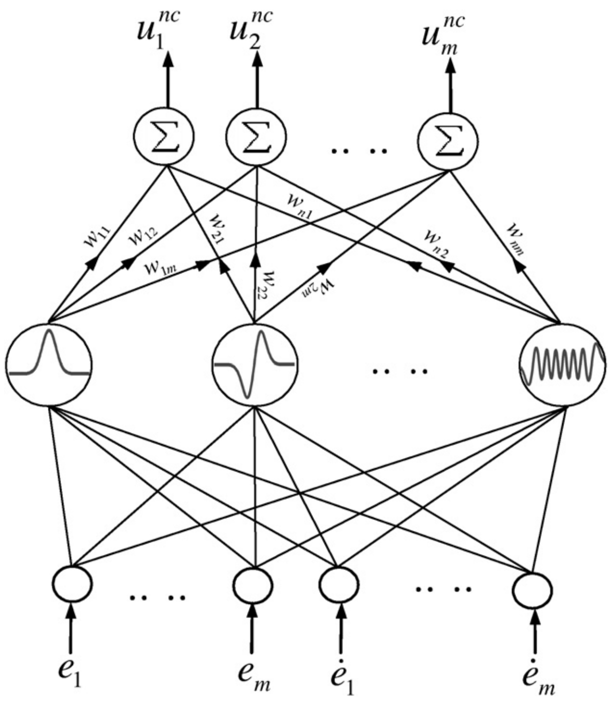

Fig. 1. The architecture of HNN.

where $\mathbf{\Lambda} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\mathbf{K}_1 & -\mathbf{K}_2 \end{bmatrix}$. Suppose the feedback gain $\underline{\mathbf{K}}$ is chosen to correspond with the coefficients of a Hurwitz polynomial, it implies that $\lim_{t\to\infty} \left\|\underline{\mathbf{e}}\right\| = 0$ for any starting initial conditions. However, since the uncertainties $\mathbf{z}(\underline{\mathbf{x}}, t)$ are always unknown or perturbed in practical applications, the ideal controller (5) cannot be precisely obtained. A trade-off problem between chattering and control accuracy arises in the ideal control scheme.

3. Description of HNN

The proposed HNN is shown in Fig. 1 is composed of the input layer, the hidden layer and the output layer. In this study, the orthogonal Hermite polynomial basis functions is adopted as the activation function for each hidden neuron and is represented as [12]

$$\Phi_i = \frac{1}{\sqrt{2^i i! \sqrt{\pi}}} e^{-\theta_j^2/2} H_i(\theta), \quad \text{for } i = 1, 2, \ldots, n \tag{7}$$

where the orthogonal Hermite polynomials are given recursively by $H_1(\theta) = 1,\quad H_2(\theta) = 2\theta,\quad \ldots, H_n(\theta) = 2\theta H_{n-1}(\theta) - 2(n-1)H_{n-2}(\theta)$ for $n \geq 3$. In addition, the input of the orthogonal Hermite polynomials basis function can be represented as

$$\theta = \sum_{j=1}^{m} e_j + \dot{e}_j. \tag{8}$$

Thus, the kth output of HNN can be expressed as

$$u_k^{nc} = \sum_{i=1}^{n} w_{ik}\Phi_i = \mathbf{w}_k^T\mathbf{\Phi}, \quad \text{for } k = 1, 2, \ldots, m \tag{9}$$

where w_{ik} denotes the connecting weight value of the kth output layer with the ith hidden layer, $\mathbf{w}_k = [w_{1k}, w_{2k}, \ldots, w_{nk}]^T$ and $\mathbf{\Phi} = [\Phi_1, \Phi_2, \ldots, \Phi_n]^T$. For ease of notation, (9) can be rewritten in a vector form as

$$\mathbf{u}^{nc} = \mathbf{W}^T\mathbf{\Phi} \tag{10}$$

where $\mathbf{u}^{nc} = [u_1^{nc}, u_2^{nc}, \ldots, u_m^{nc}]^T$ and $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_m]$.

The main property of neural networks regarding feedback control purpose is the universal function approximation property. It implies that there exists an optimal HNN $\mathbf{u}^{nc*} = [u_1^{nc*}, u_2^{nc*}, \ldots, u_m^{nc*}]^T$ such that [14,15]

$$\mathbf{u}^* = \mathbf{u}^{nc*} + \mathbf{\Delta} = \mathbf{W}^{*T}\mathbf{\Phi} + \mathbf{\Delta} \tag{11}$$

where $\mathbf{\Delta}$ denotes the approximation error vector, $\mathbf{W}^* = [\mathbf{w}_1^*, \mathbf{w}_2^*, \ldots, \mathbf{w}_m^*]$ is the optimal parameter matrix of $\mathbf{W}$. In fact, the optimal parameters that are needed to best approximate an ideal controller $\mathbf{u}^*$ are difficult to determine. Thus, an estimated HNN $\hat{\mathbf{u}}^{nc} = [\hat{u}_1^{nc}, \hat{u}_2^{nc}, \ldots, \hat{u}_m^{nc}]^T$ is defined as

$$\hat{\mathbf{u}}^{nc} = \hat{\mathbf{W}}^T\mathbf{\Phi} \tag{12}$$

where $\hat{\mathbf{W}} = [\hat{\mathbf{w}}_1, \hat{\mathbf{w}}_2, \ldots, \hat{\mathbf{w}}_m]$ is the estimated parameter matrix of $\mathbf{W}$. Define the estimation error as

$$\tilde{\mathbf{u}} = \boldsymbol{u}^* - \hat{\mathbf{u}}^{nc} = \boldsymbol{W}^{*T}\boldsymbol{\Phi} - \hat{\mathbf{W}}^T\boldsymbol{\Phi} + \boldsymbol{\Delta} = \tilde{\mathbf{W}}^T\boldsymbol{\Phi} + \boldsymbol{\Delta} \tag{13}$$

where $\tilde{\mathbf{W}} = \boldsymbol{W}^* - \hat{\mathbf{W}}$. To speed up the parameter convergence of HNN, the optimal parameter matrix $\mathbf{W}^*$ is decomposed into two parts as [26,27]

$$\mathbf{W}^* = \eta_P\mathbf{W}_\mathbf{P}^* + \eta_I\mathbf{W}_\mathbf{I}^* \tag{14}$$

where $\mathbf{W}_P^* = [\mathbf{w}_1^{\mathbf{P}*}, \mathbf{w}_2^{\mathbf{P}*}, \ldots, \mathbf{w}_m^{\mathbf{P}*}]$ and $\mathbf{W}_I^* = [\mathbf{w}_1^{\mathbf{I}*}, \mathbf{w}_2^{\mathbf{I}*}, \ldots, \mathbf{w}_m^{\mathbf{I}*}]$ are the proportional and integral terms of $\mathbf{W}^*$, respectively, η_P and η_I are positive constants, and $\mathbf{W}_\mathbf{I}^* = \int_0^t \mathbf{W}_\mathbf{P}^*(\tau)\,d\tau$. Similarly, the estimation parameter matrix $\hat{\mathbf{W}}$ is decomposed into two parts as [26,27]

$$\hat{\mathbf{W}} = \eta_P\hat{\mathbf{W}}_P + \eta_I\hat{\mathbf{W}}_I \tag{15}$$

where $\hat{\mathbf{W}}_P = [\hat{\mathbf{w}}_1^\mathbf{P}, \hat{\mathbf{w}}_2^\mathbf{P}, \ldots, \hat{\mathbf{w}}_m^\mathbf{P}]$ and $\hat{\mathbf{W}}_\mathbf{I} = [\hat{\mathbf{w}}_1^\mathbf{I}, \hat{\mathbf{w}}_2^\mathbf{I}, \ldots, \hat{\mathbf{w}}_m^\mathbf{I}]$ are the proportional and integral terms of $\hat{\mathbf{W}}$, respectively, and $\hat{\mathbf{W}}_\mathbf{I} = \int_0^t \hat{\mathbf{W}}_\mathbf{P}(\tau)\,d\tau$. Thus, $\tilde{\mathbf{W}}$ can be re-expressed as

$$\tilde{\mathbf{W}} = \eta_I\tilde{\mathbf{W}}_\mathbf{I} - \eta_P\hat{\mathbf{W}}_\mathbf{P} + \eta_P\mathbf{W}_\mathbf{P}^* \tag{16}$$

where $\tilde{\mathbf{W}}_\mathbf{I} = \mathbf{W}_\mathbf{I}^* - \hat{\mathbf{W}}_\mathbf{I} = [\tilde{\mathbf{w}}_1^\mathbf{I}, \tilde{\mathbf{w}}_2^\mathbf{I}, \ldots, \tilde{\mathbf{w}}_m^\mathbf{I}]$. Substituting (16) into (13) yields

$$\tilde{\mathbf{u}} = (\eta_I\tilde{\mathbf{W}}_I - \eta_P\hat{\mathbf{W}}_\mathbf{P} + \eta_P\mathbf{W}_\mathbf{P}^*)^T\mathbf{\Phi} + \mathbf{\Delta} = \eta_I\tilde{\mathbf{W}}_\mathbf{I}^T\mathbf{\Phi} - \eta_P\hat{\mathbf{W}}_\mathbf{P}^T\mathbf{\Phi} + \boldsymbol{\varepsilon} \tag{17}$$

where $\boldsymbol{\varepsilon} = \eta_P\mathbf{W}_\mathbf{P}^{*T}\mathbf{\Phi} + \mathbf{\Delta}$ denotes the lump of approximation error but it cannot be obtained in practice.

4. APIHNC system design

The proposed APIHNC system is shown in Fig. 2, where the control law is designed as

$$\mathbf{u}_{ac} = \hat{\mathbf{u}}^{nc} + \mathbf{u}^{rc}. \tag{18}$$

The neural controller $\hat{\mathbf{u}}^{nc}$ uses an HNN to online approximate the ideal controller and the robust compensator $\mathbf{u}^{rc}$ is designed to eliminate the effect of the approximation error introduced by the neural controller upon the system stability. Substituting (18) into (3) and using (5), the error dynamic equation can be obtained as

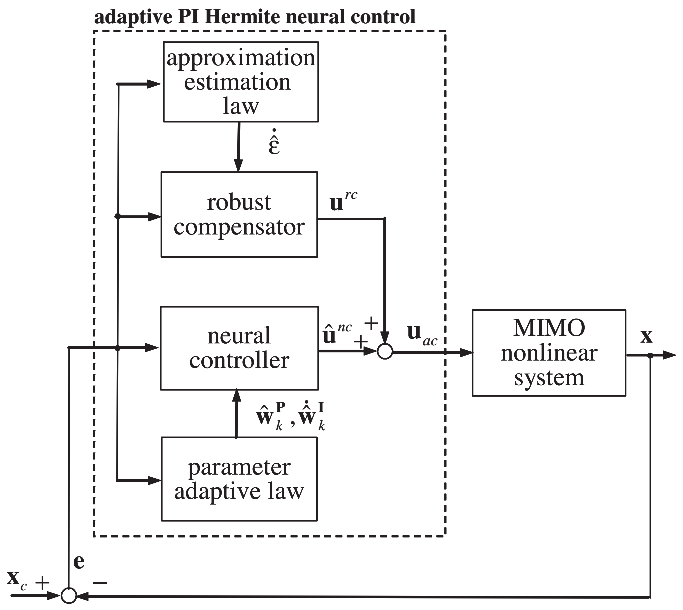

Fig. 2. The block diagram of the proposed APIHNC system.

$$\dot{\underline{\mathbf{e}}} = \mathbf{\Lambda}\underline{\mathbf{e}} + \mathbf{B}(\mathbf{u}^* - \hat{\mathbf{u}}^{nc} - \mathbf{u}^{rc}) \tag{19}$$

where $\mathbf{B} = [\mathbf{0}, \mathbf{G}_n(\underline{\mathbf{x}}, t)]^T$. Using the approximation ability of HNN, (19) can be rewritten as

$$\dot{\underline{\mathbf{e}}} = \mathbf{\Lambda}\underline{\mathbf{e}} + \mathbf{B}(\eta_I\tilde{\mathbf{W}}_\mathbf{I}^T\mathbf{\Phi} - \eta_P\hat{\mathbf{W}}_\mathbf{P}^T\mathbf{\Phi} + \boldsymbol{\varepsilon} - \mathbf{u}^{rc}). \tag{20}$$

For guaranteeing the system stability, this paper designs a robust compensator as

$$\mathbf{u}^{rc} = \hat{\boldsymbol{\varepsilon}} + \delta\mathbf{B}^T\mathbf{P}\underline{\mathbf{e}} \tag{21}$$

where $\hat{\boldsymbol{\varepsilon}}$ denotes the estimated value of the lumped approximation error, δ is a small positive constant and $\mathbf{P}$ is a symmetric positive definite matrix that satisfies the Lyapunov equation

$$\mathbf{\Lambda}^T\mathbf{P} + \mathbf{P}^T\mathbf{\Lambda} = -\mathbf{Q} \tag{22}$$

and $\mathbf{Q}$ is a positive definite matrix. Substituting (21) into (20) yields

$$\dot{\underline{\mathbf{e}}} = \mathbf{\Lambda}\underline{\mathbf{e}} + \mathbf{B}(\eta_I\tilde{\mathbf{W}}_\mathbf{I}^T\mathbf{\Phi} - \eta_P\hat{\mathbf{W}}_\mathbf{P}^T\mathbf{\Phi} + \tilde{\boldsymbol{\varepsilon}} - \delta\mathbf{B}^T\mathbf{P}\underline{\mathbf{e}}) \tag{23}$$

where $\tilde{\boldsymbol{\varepsilon}} = \boldsymbol{\varepsilon} - \hat{\boldsymbol{\varepsilon}}$. To guarantee the stability of the APIHNC system, consider a Lyapunov function candidate in the following form as

$$V = \frac{1}{2}\underline{\mathbf{e}}^T\mathbf{P}\underline{\mathbf{e}} + \frac{\eta_I}{2}tr(\tilde{\mathbf{W}}_\mathbf{I}^T\tilde{\mathbf{W}}_\mathbf{I}) + \frac{1}{2\eta_\varepsilon}\tilde{\boldsymbol{\varepsilon}}^T\tilde{\boldsymbol{\varepsilon}} \tag{24}$$

where η_ε is a positive learning rate. Taking the derivative of Lyapunov function in (24) and using (23), yields

$$\begin{aligned}
\dot{V} &= \frac{1}{2}\dot{\underline{\mathbf{e}}}^T\boldsymbol{P}\underline{\boldsymbol{e}} + \frac{1}{2}\underline{\boldsymbol{e}}^T\boldsymbol{P}\dot{\underline{\mathbf{e}}} + \eta_I tr(\tilde{\mathbf{W}}_\boldsymbol{I}^T\dot{\tilde{\mathbf{W}}}_\boldsymbol{I}) + \frac{1}{\eta_\varepsilon}\tilde{\varepsilon}^T\dot{\tilde{\varepsilon}} \\
\dot{V} &= -\frac{1}{2}\underline{\mathbf{e}}^T\mathbf{Q}\underline{\mathbf{e}} + \underline{\mathbf{e}}^T\mathbf{PB}(\eta_I\tilde{\mathbf{W}}_I^T\mathbf{\Phi} - \eta_P\hat{\mathbf{W}}_\mathbf{P}^T\mathbf{\Phi} + \tilde{\boldsymbol{\varepsilon}} - \delta\mathbf{B}^T\mathbf{P}\underline{\mathbf{e}}) + \eta_I tr(\tilde{\mathbf{W}}_I^T\dot{\tilde{\mathbf{W}}}_I) + \frac{1}{\eta_\varepsilon}\tilde{\boldsymbol{\varepsilon}}^T\dot{\hat{\boldsymbol{\varepsilon}}} \\
\dot{V} &= -\frac{1}{2}\underline{\mathbf{e}}^T\mathbf{Q}\underline{\mathbf{e}} + \eta_I\underline{\mathbf{e}}^T\mathbf{PB}\tilde{\mathbf{W}}_\mathbf{I}^T\mathbf{\Phi} - \eta_P\underline{\mathbf{e}}^T\mathbf{PB}\hat{\mathbf{W}}_P^T\mathbf{\Phi} + \underline{\mathbf{e}}^T\mathbf{PB}\tilde{\varepsilon} - \delta\underline{\mathbf{e}}^T\mathbf{PBB}^T\mathbf{P}\underline{\mathbf{e}} + \eta_I tr(\tilde{\mathbf{W}}_\boldsymbol{I}^T\dot{\tilde{\mathbf{W}}}_\boldsymbol{I}) + \frac{1}{\eta_\varepsilon}\tilde{\varepsilon}^T\dot{\tilde{\varepsilon}} \\
\dot{V} &= -\frac{1}{2}\underline{\boldsymbol{e}}^T\boldsymbol{Q}\underline{\boldsymbol{e}} + \eta_I\underline{\boldsymbol{e}}^T\boldsymbol{PB}\tilde{\mathbf{W}}_\boldsymbol{I}^T\boldsymbol{\Phi} - \eta_P\underline{\boldsymbol{e}}^T\boldsymbol{PB}\hat{\mathbf{W}}_\boldsymbol{P}^T\boldsymbol{\Phi} + \tilde{\varepsilon}^T\boldsymbol{B}^T\boldsymbol{P}\underline{\boldsymbol{e}} - \delta\underline{\boldsymbol{e}}^T\boldsymbol{PBB}^T\boldsymbol{P}\underline{\boldsymbol{e}} + \eta_I tr(\tilde{\mathbf{W}}_\boldsymbol{I}^T\dot{\tilde{\mathbf{W}}}_\boldsymbol{I}) + \frac{1}{\eta_\varepsilon}\tilde{\varepsilon}^T\dot{\tilde{\varepsilon}}
\end{aligned} \tag{25}$$

where $\underline{\mathbf{e}}^T\mathbf{PB}\tilde{\boldsymbol{\varepsilon}} = \tilde{\boldsymbol{\varepsilon}}^T\mathbf{B}^T\mathbf{P}\underline{\mathbf{e}}$ is used since it is a scale. Chooses $\hat{\mathbf{w}}_k^\mathbf{P}$ as

$$\hat{\mathbf{w}}_k^\mathbf{P} = \underline{\mathbf{e}}^T\mathbf{P}\mathbf{b}_k\mathbf{\Phi} \tag{26}$$

and noting that

$$\underline{\mathbf{e}}^T\mathbf{PB}\tilde{\mathbf{W}}_{\mathbf{I}}^T\mathbf{\Phi} = \sum_{k=1}^{m}\underline{\mathbf{e}}^T\mathbf{Pb}_k\tilde{\mathbf{w}}_k^{\mathbf{I}T}\mathbf{\Phi} = \sum_{k=1}^{m}\tilde{\mathbf{w}}_k^{\mathbf{I}T}\underline{\mathbf{e}}^T\mathbf{Pb}_k\mathbf{\Phi} \tag{27}$$

$$\underline{\mathbf{e}}^T\mathbf{PB}\hat{\mathbf{W}}_{\mathbf{P}}^T\mathbf{\Phi} = \sum_{k=1}^{m}\underline{\mathbf{e}}^T\mathbf{Pb}_k\hat{\mathbf{w}}_k^{\mathbf{P}T}\mathbf{\Phi} = \sum_{k=1}^{m}\hat{\mathbf{w}}_k^{\mathbf{P}T}\underline{\mathbf{e}}^T\mathbf{Pb}_k\mathbf{\Phi} \tag{28}$$

$$tr(\tilde{\mathbf{W}}_{\mathbf{I}}^T\dot{\tilde{\mathbf{W}}}_{\mathbf{I}}) = \sum_{k=1}^{m}\tilde{\mathbf{w}}_k^{\mathbf{I}T}\dot{\tilde{\mathbf{w}}}_k^{\mathbf{I}} \tag{29}$$

where $\mathbf{b}_k$ is the kth column of matrix $\mathbf{B}$ and $\underline{\mathbf{e}}^T\mathbf{Pb}_k$ is a scale, thus (25) can be obtained as

$$\begin{aligned}\dot{V} &= -\frac{1}{2}\underline{\mathbf{e}}^T\mathbf{Q}\underline{\mathbf{e}} + \eta_I\sum_{k=1}^{m}\tilde{\mathbf{w}}_k^{\mathbf{I}T}(\underline{\mathbf{e}}^T\mathbf{Pb}_k\mathbf{\Phi} + \dot{\tilde{\mathbf{w}}}_k^{\mathbf{I}}) \\ &\quad - \eta_P\sum_{k=1}^{m}\hat{\mathbf{w}}_k^{\mathbf{P}T}\hat{\mathbf{w}}_k^{\mathbf{P}} + \tilde{\varepsilon}^T\left(\mathbf{B}^T\mathbf{P}\underline{\mathbf{e}} + \frac{1}{\eta_\varepsilon}\dot{\tilde{\boldsymbol{\varepsilon}}}\right) - \delta\underline{\mathbf{e}}^T\mathbf{PBB}^T\mathbf{P}\underline{\mathbf{e}}\end{aligned} \tag{30}$$

If the adaptation laws are chosen as

$$\dot{\hat{\mathbf{w}}}_k^{\mathbf{I}} = -\dot{\tilde{\mathbf{w}}}_k^{\mathbf{I}} = \underline{\mathbf{e}}^T\mathbf{Pb}_k\hat{\mathbf{\Phi}} \tag{31}$$

$$\dot{\hat{\boldsymbol{\varepsilon}}} = -\dot{\tilde{\boldsymbol{\varepsilon}}} = \eta_\varepsilon\mathbf{B}^T\mathbf{P}\underline{\mathbf{e}} \tag{32}$$

thus (30) can be obtained as

$$\dot{V} = -\frac{1}{2}\underline{\mathbf{e}}^T\mathbf{Q}\underline{\mathbf{e}} - \eta_P\sum_{k=1}^{m}\hat{\mathbf{w}}_k^{\mathbf{P}T}\hat{\mathbf{w}}_k^{\mathbf{P}} - \delta\underline{\mathbf{e}}^T\mathbf{PBB}^T\mathbf{P}\underline{\mathbf{e}} \le -\frac{1}{2}\underline{\mathbf{e}}^T\mathbf{Q}\underline{\mathbf{e}} \le 0. \tag{33}$$

Since $\dot{V} \le 0$ is a negative semi-definite function, which implies $\underline{\mathbf{e}}$, $\tilde{\mathbf{W}}_{\mathbf{I}}$, and $\tilde{\boldsymbol{\varepsilon}}$ are bounded. Let function $\Omega(t) = (1/2)\underline{\mathbf{e}}^T\mathbf{Q}\underline{\mathbf{e}} \le -\dot{V}$, and integrate function $\Omega(t)$ with respect to time

$$\int_0^t \Omega(\tau)d\tau \le V(0) - V(t). \tag{34}$$

Because $V(0)$ is bounded and $V(t)$ is non-increasing and bounded, the following result can be obtained

$$\lim_{t\to\infty}\int_0^t \Omega(\tau)d\tau < \infty. \tag{35}$$

In addition, since $\dot{\Omega}(t)$ is bounded, so by Barbalat's Lemma [1], it can be shown that $\lim_{t\to\infty}\Omega(t) = 0$. This will imply that $\left\|\underline{\mathbf{e}}\right\|$ converges to zero as $t\to\infty$ As a result, the stability of the proposed APIHNC system can be guaranteed.

5. Simulation results

In this section, the proposed APIHNC system is applied to an inverted double pendulums and a two-link robotic manipulator to verify its effectiveness. It should be emphasized that the development of the proposed APIHNC system does not need to know the controlled system dynamics. For practical implementation, the controller parameters can be online tuned by the proposed adaptive laws without the need of the system parameters. A HNN is used to online mimic an ideal controller. In general, a better approximated performance can be obtained if a higher-order orthogonal Hermite polynomial basis functions is used. By choosing the values of $\mathbf{K}_1$ and $\mathbf{K}_2$ properly, the desired system dynamics such as settling time can be easily designed by the second-order system. The parameters η_P and η_I are the leaning rates of neural controller and the parameter η_ε is the leaning rate of robust compensator. If the leaning rates are chosen small, the parameter convergence will be easily achieved; however, this will result in slow learning speed. On the other hand, if the leaning rates are chosen large, the learning speed will be fast; however, the system may become unstable. $\mathbf{Q}$ and δ will influence the convergent speed of tracking error. A better control performance can be achieved as $\mathbf{Q}$ and δ are chosen larger but it leads to a large control signal.

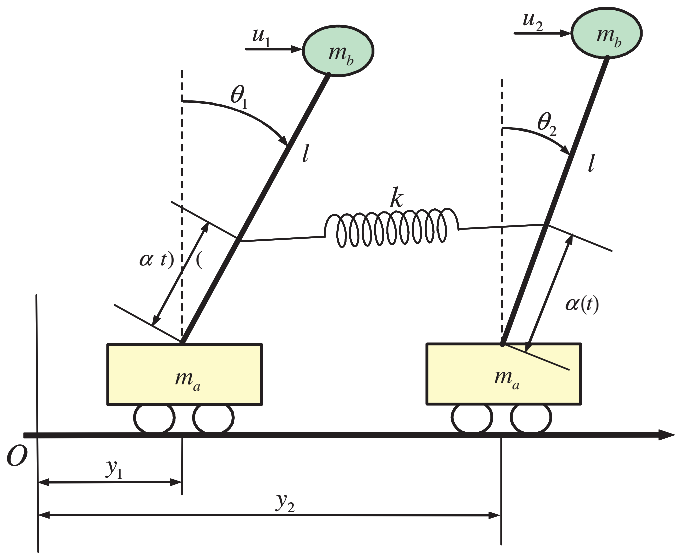

Fig. 3. Inverted double pendulums.

Example 1. Inverted double pendulums

Consider inverted double pendulums connected by a moving spring mounted on two carts as shown in Fig. 3. The pivot position of the moving spring is a function of time that can change along the length of pendulums. The system dynamics of inverted double pendulums on carts is given as follows [28,29]:

$$\begin{aligned}\ddot{\theta}_1 &= \frac{g}{c_m l}\theta_1 + \frac{1}{c_m m_b l^2}u_1 + N_1(\theta_1, \dot{\theta}_1) \\ &\quad + \left[\frac{k(\alpha(t) - c_m l)}{c_m m_b l^2}(-\alpha(t)\theta_1 + \alpha(t)\theta_2 - y_1 + y_2)\right]\end{aligned} \tag{36}$$

$$\begin{aligned}\ddot{\theta}_2 &= \frac{g}{c_m l}\theta_2 + \frac{1}{c_m m_b l^2}u_2 + N_2(\theta_2, \dot{\theta}_2) \\ &\quad + \left[\frac{k(\alpha(t) - c_m l)}{c_m m_b l^2}(-\alpha(t)\theta_2 + \alpha(t)\theta_1 - y_1 + y_2)\right]\end{aligned} \tag{37}$$

where θ_i and $\dot{\theta}_i$ are the angle and angular velocity of pendulums, u_i is the control torque applied to the pendulums, k and g denote the spring and the gravity constant, respectively, m_a and m_b are the masses of the carts and the pendulums, respectively, $c_m = m_b/(m_a + m_b)$, l is the length of the pendulums, y_1 and y_2 are the displacements of the moving carts, $\alpha(t) = \sin(5t)$ and $N_1(\theta_1, \dot{\theta}_1)$ and $N_2(\theta_2, \dot{\theta}_2)$ are system uncertainties. Define $\mathbf{x}\underline{\mathbf{\Delta}}[\theta_1, \theta_2]^T = [x_1, x_2]^T$, the dynamics (36) and (37) can be expressed as

$$\ddot{\mathbf{x}} = \mathbf{f}(\underline{\mathbf{x}}, t) + \mathbf{G}(\underline{\mathbf{x}}, t)\mathbf{u} + \mathbf{d}(\underline{\mathbf{x}}, t) \tag{38}$$

where

$$\underline{\mathbf{x}} = [\mathbf{x}^T, \dot{\mathbf{x}}^T]^T,$$

$$\mathbf{f}(\underline{\mathbf{x}}, t) = \begin{bmatrix} \dfrac{g}{c_m l}x_1 + \dfrac{k(\alpha(t) - c_m l)}{c_m m_b l^2}(-\alpha(t)x_1 + \alpha(t)x_2 - y_1 + y_2) \\ \dfrac{g}{c_m l}x_2 + \dfrac{k(\alpha(t) - c_m l)}{c_m m_b l^2}(-\alpha(t)x_2 + \alpha(t)x_1 - y_1 + y_2) \end{bmatrix},$$

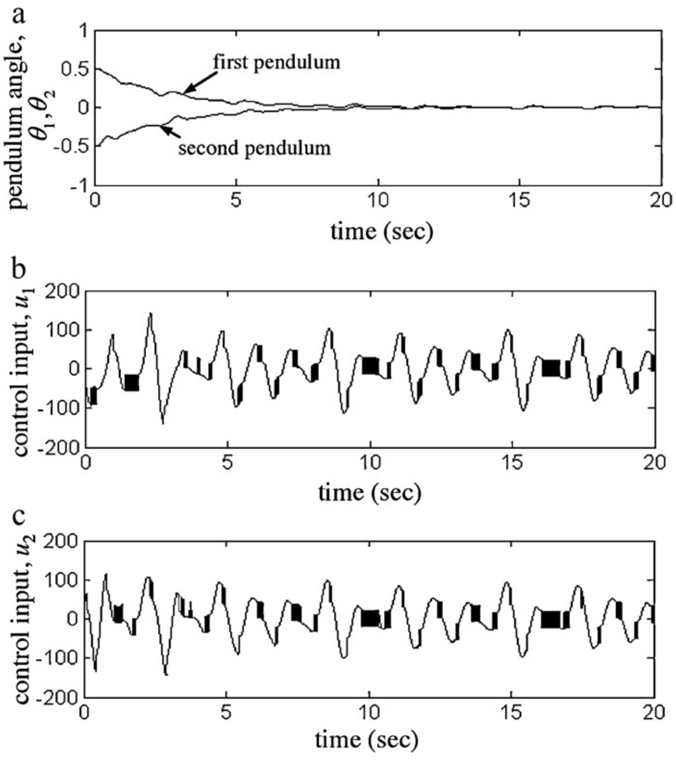

Fig. 4. Simulation results of the RCMAC hybrid control [9].

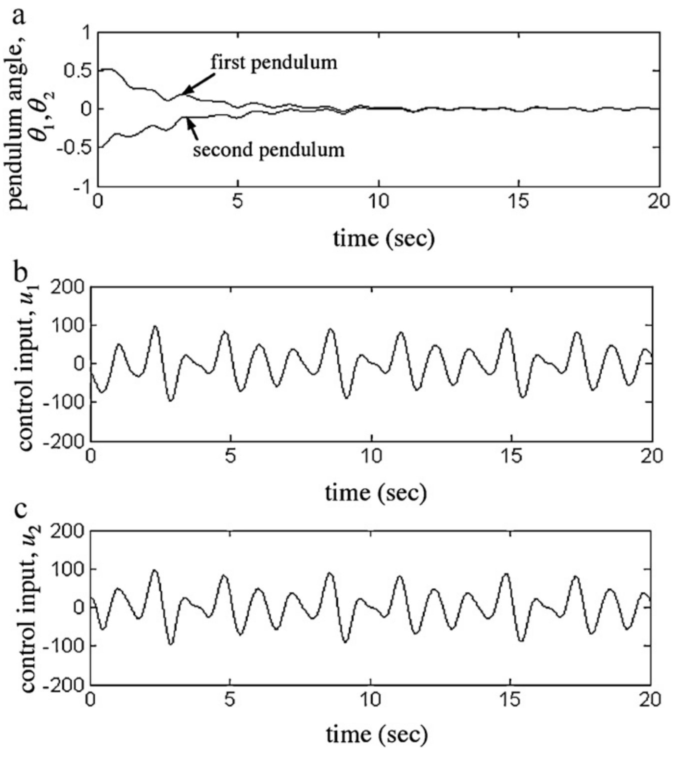

Fig. 5. Simulation results of the APIHNC system with integral learning algorithm.

$$\mathbf{G}(\underline{\mathbf{x}}, t) = \begin{bmatrix} \dfrac{1}{c_m m_b l^2} & 0 \\ 0 & \dfrac{1}{c_m m_b l^2} \end{bmatrix},$$

$$\mathbf{u} = [u_1, u_2]^T \quad \text{and} \quad \mathbf{d}(\underline{\mathbf{x}}, t) = \begin{bmatrix} N_1(x_1, \dot{x}_1) \\ N_2(x_2, \dot{x}_2) \end{bmatrix}.$$

The parameters of inverted double pendulums are $m_a = m_b = 50\,\text{kg}$, $k = 1$, $g = 1$, $l = 1$, $N_1(x_1, \dot{x}_1) = -(m_b/m_a)\dot{x}_1^2 \sin(x_1)$, $N_2(x_2, \dot{x}_2) = -(m_b/m_a)\dot{x}_2^2 \sin(x_2)$, $\alpha(t) = \sin(5t)$, $y_1 = \sin(2t)$ and $y_2 = \sin(3t) + 2$ [29].

The parameters for the APIHNC system are selected as $\mathbf{K}_1 = \begin{bmatrix} 0.04 & 0 \\ 0 & 0.04 \end{bmatrix}$, $\mathbf{K}_2 = \begin{bmatrix} 0.4 & 0 \\ 0 & 0.4 \end{bmatrix}$, $\eta_P = 5$, $\eta_I = 50$, $\eta_\varepsilon = 1$ and $\delta = 10$. The selections of these values are through some trials to achieve satisfactory control performance. In this example, a HNN with nine hidden neurons is utilized to approximate an ideal controller. For a choice of $Q = \text{diag}(20, 20, 20, 20)$, solving the Lyapunov Eq. (22), obtain

$$\mathbf{P} = \begin{bmatrix} 126 & 0 & 250 & 0 \\ 0 & 126 & 0 & 250 \\ 250 & 0 & 650 & 0 \\ 0 & 250 & 0 & 650 \end{bmatrix}. \tag{39}$$

To illustrate the effectiveness of the proposed design method, a comparison between the RCMAC hybrid control [9] and the proposed APIHNC system is made. First, the RCMAC hybrid control [9] is applied to control. The simulation results of the RCMAC hybrid control system for inverted double pendulum are shown in Fig. 4. The trajectory of system states θ_1 and θ_2 is shown in Fig. 4(a), and the control inputs u_1 and u_2 are shown in Fig. 4(b) and (c), respectively. The simulation results show that favorable control performance can be achieved but the undesirable chattering phenomenon occurs as shown in Fig. 4(b) and (c). Then, the proposed APIHNC system is applied to control again. To compare the control efficiency, the APIHNC system with integral parameter adaptation is considered. This is a special case of the developed APIHNC design method for $\eta_P = 0$. As $\eta_P = 0$, the parameter adaptation results in an integral tuning mechanism which can be found in many previous published papers [15]. The simulation results of the APIHNC system with integral parameter adaptation for inverted double pendulum are shown in Fig. 5. The trajectory of system states θ_1 and θ_2 is shown in Fig. 5(a), and the control inputs u_1 and u_2 are shown in Fig. 5(b) and (c), respectively. From the simulation results, excellent tracking performance and chattering-free in control effort are achieved for the proposed APIHNC system. But, the convergence of tracking error is slow using the integral type learning algorithm. Finally, the proportional–integral learning algorithm is applied with $\eta_P = 5$. The simulation results of the APIHNC system with proportional–integral parameter adaptation for inverted double pendulum are shown in Fig. 6. The trajectory of system states θ_1 and θ_2 is shown in Fig. 6(a), and the control inputs u_1 and u_2 are shown in Fig. 6(b) and (c), respectively. The simulation results show that the favorable tracking performance can be achieved and the tracking errors converge more quickly than that in Fig. 5.

Example 2. Two-link robotic manipulator

A two-link robotic manipulator is shown in Fig. 7. The system dynamic of a two-link robotic manipulator system can be expressed in the Lagrange following form [30–33]

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \boldsymbol{\tau} \tag{40}$$

where $\mathbf{q} = [q_1, q_2]^T$ is the joint vector of robotic manipulator, $\mathbf{M}(\mathbf{q})$ denotes the inertia matrix, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ expresses the matrix of centripetal and Coriolis forces, $\mathbf{G}(\mathbf{q})$ is the gravity vector, and $\boldsymbol{\tau}$ is the

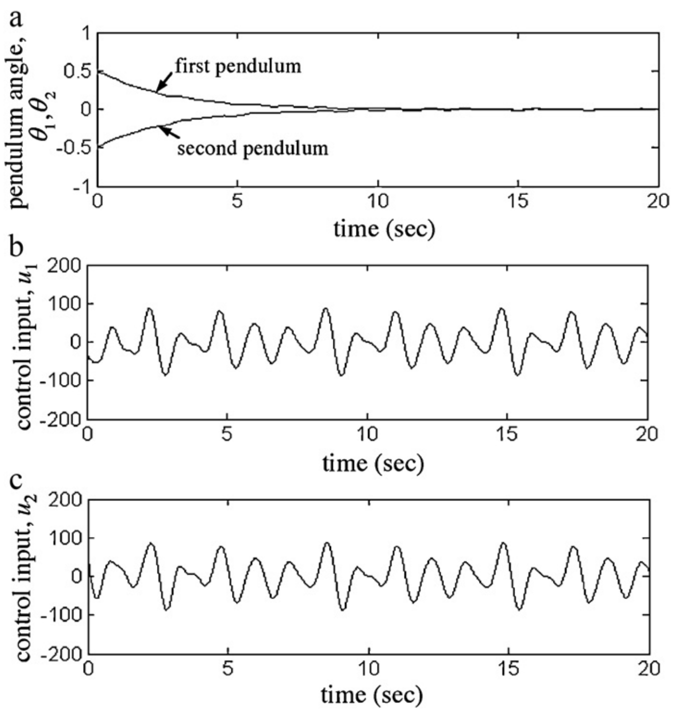

Fig. 6. Simulation results of the APIHNC system with proportional–integral learning algorithm.

torque vectors exerting on joints. The system (40) can be transformed into the form of (1) with

$$\ddot{\mathbf{q}} = \begin{bmatrix} f_1(\underline{\mathbf{q}}, t) \\ f_2(\underline{\mathbf{q}}, t) \end{bmatrix} + \begin{bmatrix} g_{11}(\underline{\mathbf{q}}, t) & g_{12}(\underline{\mathbf{q}}, t) \\ g_{21}(\underline{\mathbf{q}}, t) & g_{22}(\underline{\mathbf{q}}, t) \end{bmatrix} \mathbf{u} \tag{41}$$

$$f_1(\underline{\mathbf{q}}, t) = M_2 l_1 l_2 a_1 a_3 q_1^2 - M_2 l_2^2 a_1 q_2^2 + \frac{(M_1 + M_2) g s_1 - M_2 g a_3 s_2}{l_1 a_2} \tag{42}$$

$$f_2(\underline{\mathbf{q}}, t) = M_2 l_1 l_2 a_1 a_3 q_2^2 + (M_1 + M_2) \left(l_1^2 a_1 q_1^2 - \frac{g a_3 s_1 + g s_2}{l_2 a_2} \right) \tag{43}$$

$$g_{11}(\underline{\mathbf{q}}, t) = \frac{1}{l_1^2 a_2} \tag{44}$$

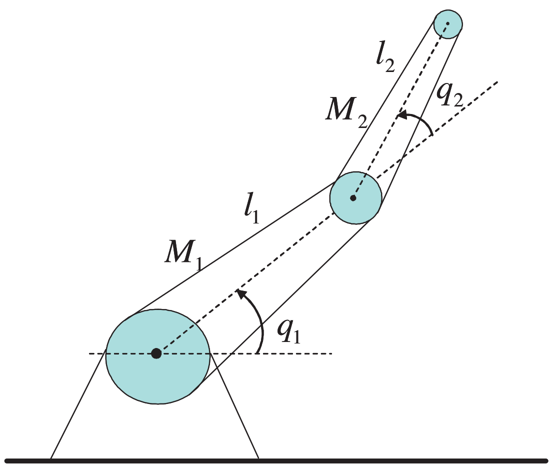

Fig. 7. Two-link robotic manipulator system.

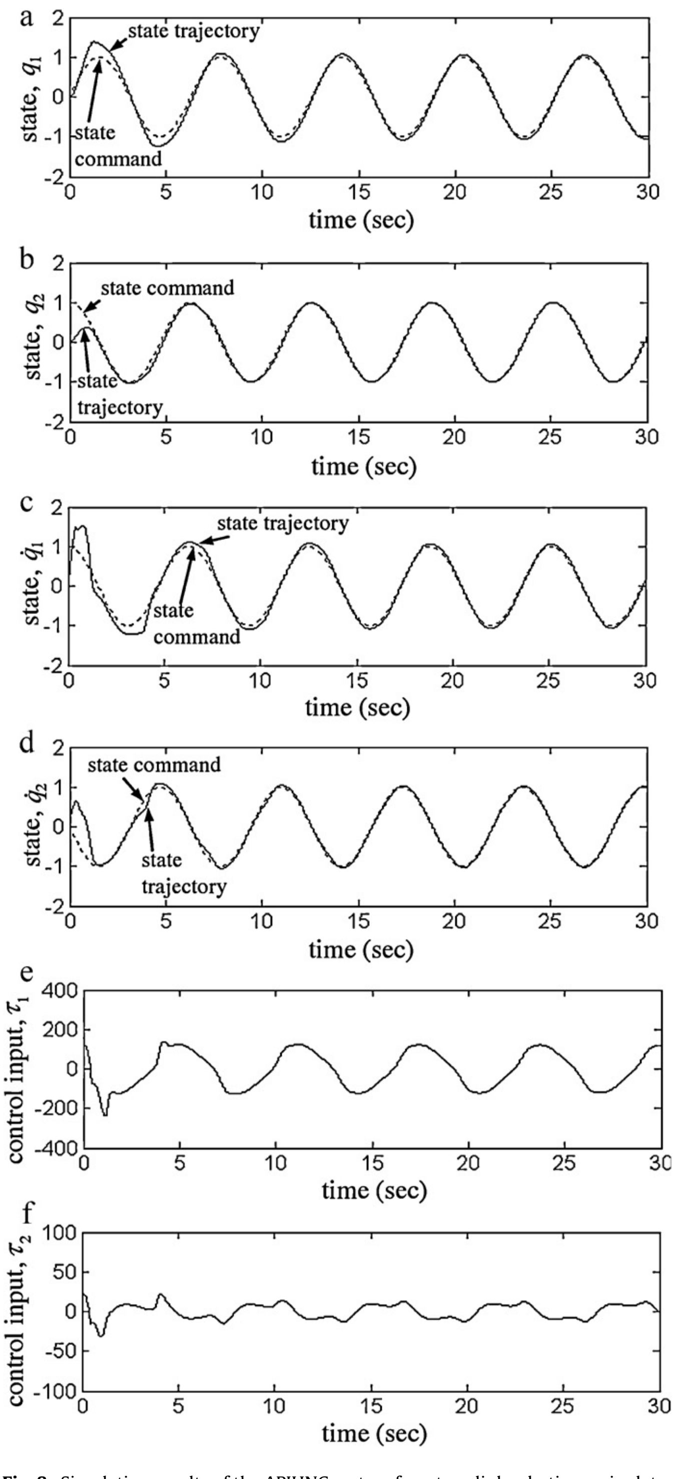

Fig. 8. Simulation results of the APIHNC system for a two-link robotic manipulator system.

$$g_{12}(\underline{\mathbf{q}}, t) = g_{21}(\underline{\mathbf{q}}, t) = \frac{-a_3}{l_1 l_2 a_2} \tag{45}$$

$$g_{22}(\underline{\mathbf{q}}, t) = \frac{M_1 + M_2}{M_1 l_2^2 a_2} \tag{46}$$

where $\underline{\mathbf{q}} = [\mathbf{q}^T, \dot{\mathbf{q}}^T]^T$, M_1 and M_2 are link masses, l_1 and l_2 are link lengths, g is acceleration, $s_1 = \sin(q_1)$, $s_2 = \sin(q_2)$, $c_1 = \cos(q_1)$, $c_2 = \cos(q_2)$, $a_1 = (s_1 c_2 - c_1 s_2)/a_2$, $a_2 = M_1 + M_2 - M_2 a_3^2$, and $a_3 = c_1 c_2 + s_1 s_2$. For the simulation, the system parameter are given as $M_1 = 4$ kg, $M_2 = 2$ kg, $l_1 = 2$ m, $l_2 = 1$ m, and $g = 9.8$ m/s^2

Table 1
Comparison of controller characteristics.

Controller	Controller parameters	Robustness	Convergence speed	Chattering phenomena
RCMAC hybrid control [9]	On-line learning	Excellent	Fast	Yes
APIHNC with integral parameter adaptation	On-line learning	Middle	Slow	No
APIHNC with proportional–integral parameter adaptation	On-line learning	Excellent	Fast	No

[30]. The proposed APIHNC system is applied to control a two-link robotic manipulator. The control parameters of the APIHNC system are selected as $\boldsymbol{K}_1 = \begin{bmatrix} 0.25 & 0 \\ 0 & 0.25 \end{bmatrix}$, $\mathbf{K}_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $\eta_I = 25$, $\eta_P = \eta_\varepsilon = 1$, and $\delta = 10$. The selections of these values are through some trials to achieve satisfactory control performance. In this example, a HNN with seven hidden neurons is utilized to approximate an ideal controller. For a choice of $Q = \text{diag}(100, 25, 20, 5)$, solving the Lyapunov Eq. (22), obtain

$$\mathbf{P} = \begin{bmatrix} 252 & 0 & 200 & 0 \\ 0 & 63.125 & 0 & 50 \\ 200 & 0 & 210 & 0 \\ 0 & 50 & 0 & 52.5 \end{bmatrix}. \tag{47}$$

The simulation result of the proposed APIHNC system is shown in Fig. 8. The tracking responses of states q_1 and q_2 are shown in Fig. 8(a) and (b), respectively, the tracking responses of states $\dot{q}_1$ and $\dot{q}_2$ are shown in Fig. 8(c) and (d), respectively, and the control inputs u_1 and u_2 are shown in Fig. 8(e) and (f), respectively. The simulation results show that the favorable tracking performance can be achieved by the proposed APIHNC system without requiring a preliminary offline tuning. To demonstrate the control performance of the proposed control system with different reference trajectories, the reference trajectory commands for joints 1 and 2 are examined $0.7\sin(1.5t) + 0.3\cos(3.5t)$ and $0.7\cos(1.5t) + 0.3\sin(3.5t)$, respectively. The simulation result of the proposed APIHNC system with different reference trajectories are shown in Fig. 9. The tracking responses of states q_1 and q_2 are shown in Fig. 9(a) and (b), respectively, the tracking responses of states $\dot{q}_1$ and $\dot{q}_2$ are shown in Fig. 9(c) and (d), respectively, and the control inputs u_1 and u_2 are shown in Fig. 9(e) and (f), respectively. The simulation results show that the proposed APIHNC system can still achieve satisfactory tracking responses in the presence of different reference trajectories.

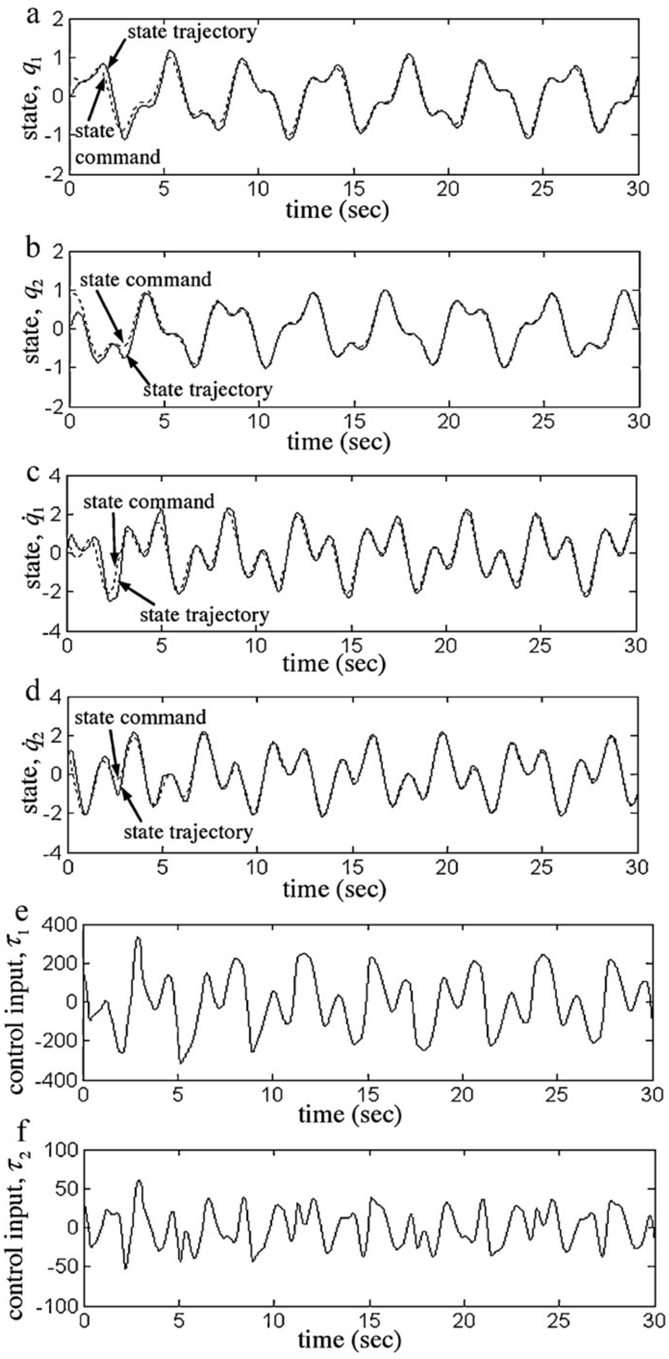

Fig. 9. Simulation results of the APIHNC system with different reference trajectories.

6. Conclusion

This paper has successfully developed an adaptive PI Hermite neural control (APIHNC) system for an inverted double pendulums and a two-link robotic manipulator. The proposed APIHNC system is composed of a neural controller and a robust compensator. The neural controller uses a Hermite neural network to online mimic an ideal controller and the robust compensator is design to efficiently suppress the influence of approximation error introduced by the neural controller upon the system stability in the Lyapunov sense. Finally, the proposed APIHNC system is applied to an inverted double pendulums and a two-link robotic manipulator. From the simulation results, it is verified that the proposed APIHNC scheme can achieve a favorable tracking performance. It possesses the salient advantages of free from chattering, stable tracking control performance, and robust to system uncertainties. In summary, the comparison of various controller characteristics is made in Table 1. It shows that the proposed APIHNC system with proportional–integral parameter adaptation is suitable for MIMO uncertain nonlinear systems.

Acknowledgments

The authors are grateful to the reviewers for their valuable comments. The authors appreciate the partial financial support from the National Science Council of Republic of China under grant NSC 100-2628-E-032-003.

References

[1] J.J.E. Slotine, W.P. Li, Applied Nonlinear Control, Prentice Hall, Englewood Cliffs, NJ, 1991.

[2] K.H. Cheng, CMAC-based neuro-fuzzy approach for complex system modeling, Neurocomputing 72 (7) (2009) 1763–1774.

[3] M.F. Yeh, C.H. Tsai, Standalone CMAC control system with online learning ability, IEEE Transactions on Systems, Man, and Cybernetics: Part B 40 (1) (2010) 43–53.

[4] T. Orlowska-Kowalska, M. Dybkowski, K. Szabat, Adaptive sliding-mode neuro-fuzzy control of the two-mass induction motor drive without mechanical sensors, IEEE Transactions on Industrial Electronics 57 (2) (2010) 553–564.

[5] C.H. Chiu, The design and implementation of a wheeled inverted pendulum using an adaptive output recurrent cerebellar model articulation controller, IEEE Transactions on Industrial Electronics 57 (5) (2010) 1814–1822.

[6] C.K. Lin, Radial basis function neural network-based adaptive critic control of induction motors, Applied Soft Computing 11 (3) (2011) 3066–3074.

[7] C.F. Hsu, Adaptive dynamic RBF neural controller design for a class of nonlinear systems, Applied Soft Computing 11 (8) (2011) 4607–4613.

[8] C.F. Hsu, Intelligent tracking control of a DC motor driver using self-organizing TSK type fuzzy neural networks, Nonlinear Dynamics 67 (1) (2012) 587–600.

[9] C.M. Lin, L.Y. Chen, C.H. Chen, RCMAC hybrid control for MIMO uncertain nonlinear systems using sliding-mode technology, IEEE Transactions on Neural Networks 18 (3) (2007) 708–720.

[10] C.S. Chen, Dynamic structure adaptive neural fuzzy control for MIMO uncertain nonlinear systems, Information Sciences 179 (15) (2009) 2676–2688.

[11] S. Aloui, O. Pagès, A. El Hajjaji, A. Chaari, Y. Koubaa, Improved fuzzy sliding mode control for a class of MIMO nonlinear uncertain and perturbed systems, Applied Soft Computing 11 (1) (2011) 820–826.

[12] M. Lagerholm, C. Peterson, G. Braccini, L. Edenbrandt, L. Sornmo, Clustering ECG complexes using Hermite functions and self-organizing maps, IEEE Transactions on Biomedical Engineering 47 (7) (2000) 838–848.

[13] T.H. Linh, S. Osowski, M. Stodolski, On-line heart beat recognition using Hermite polynomials and neuro-fuzzy network, IEEE Transactions on Instrumentation and Measurement 52 (4) (2003) 1224–1231.

[14] L. Ma, K. Khorasani, Constructive feedforward neural networks using Hermite polynomial activation functions, IEEE Transactions on Neural Networks 16 (4) (2005) 821–833.

[15] F.J. Lin, S.Y. Chen, M.S. Huang, Tracking control of thrust active magnetic bearing system via Hermite polynomial-based recurrent neural network, IET Electric Power Applications 4 (9) (2010) 701–714.

[16] J.H. Park, S.J. Seo, G.T. Park, Robust adaptive fuzzy controller for nonlinear system using estimation of bounds for approximation errors, Fuzzy Sets and Systems 133 (1) (2003) 19–36.

[17] C.M. Lin, C.F. Hsu, Supervisory recurrent fuzzy neural network control of wing rock for slender delta wings, IEEE Transactions on Fuzzy Systems 12 (5) (2004) 733–742.

[18] C.M. Lin, C.F. Hsu, Neural network hybrid control for antilock braking systems, IEEE Transactions on Neural Networks 14 (2) (2003) 351–359.

[19] F.J. Lin, D.H. Wang, P.K. Huang, FPGA-based fuzzy sliding-mode control for a linear induction motor drive, IEE Proceedings – Electric Power Applications 152 (5) (2005) 1137–1148.

[20] R.J. Wai, Fuzzy sliding-mode control using adaptive tuning technique, IEEE Transactions on Industrial Electronics 54 (1) (2007) 586–594.

[21] C.F. Hsu, K.H. Cheng, T.T. Lee, Robust wavelet-based adaptive neural controller design with a fuzzy compensator, Neurocomputing 73 (1) (2009) 423–431.

[22] C.S. Tseng, A novel approach to H_∞ decentralized fuzzy-observer-based fuzzy control design for nonlinear interconnected systems, IEEE Transactions on Fuzzy Systems 16 (5) (2008) 1337–1350.

[23] Y.F. Peng, Development of robust intelligent tracking control system for uncertain nonlinear systems using H^∞ control technique, Applied Soft Computing 11 (3) (2011) 3135–3146.

[24] C.F. Hsu, Adaptive dynamic CMAC neural control of nonlinear chaotic systems with L_2 tracking performance, Engineering Applications of Artificial Intelligence 25 (5) (2012) 997–1008.

[25] R. Shahnazi, H.M. Shanechi, N. Pariz, Position control of induction and DC servomotors: a novel adaptive fuzzy PI sliding mode control, IEEE Transactions on Energy Conversion 23 (1) (2008) 138–147.

[26] N. Golea, A. Golea, K. Benmahammed, Fuzzy model reference adaptive control, IEEE Transactions on Fuzzy Systems 10 (4) (2002) 436–444.

[27] C.F. Hsu, C.M. Chung, C.M. Lin, C.Y. Hsu, Adaptive CMAC neural control of chaotic systems with a PI-type learning algorithm, Expert Systems with Applications 36 (9) (2009) 11836–11843.

[28] S. Zhou, G. Feng, C.B. Feng, Robust control for a class of uncertain nonlinear systems: adaptive fuzzy approach based on backstepping, Fuzzy Sets and Systems 151 (1) (2005) 1–20.

[29] C.H. Chen, C.F. Hsu, Recurrent wavelet neural backstepping controller design with a smooth compensator, Neural Computing & Applications 19 (7) (2010) 1089–1100.

[30] A. Chatterjee, K. Watanabe, An optimized Takagi–Sugeno type neuro-fuzzy system for modeling robot manipulators, Neural Computing & Applications 15 (1) (2006) 55–61.

[31] N. Goléa, A. Goléa, K. Barra, T. Bouktir, Observer-based adaptive control of robot manipulators: fuzzy systems approach, Applied Soft Computing 8 (1) (2008) 778–787.

[32] W.Y. Wang, Y.H. Chien, Y.G. Leu, T.T. Lee, On-line adaptive T–S fuzzy-neural control for a class of general multi-link robot manipulators, International Journal of Fuzzy Systems 10 (4) (2008) 240–249.

[33] C.K. Lin, H_∞ reinforcement learning control of robot manipulators using fuzzy wavelet networks, Fuzzy Sets and Systems 160 (12) (2009) 1765–1786.