Neurocomputing 125 (2014) 46–56

Contents lists available at ScienceDirect

Neurocomputing

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

# Neural-network-based optimal tracking control scheme for a class of unknown discrete-time nonlinear systems using iterative ADP algorithm

Yuzhu Huang, Derong Liu *

*State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China*

ARTICLE INFO

Available online 28 February 2013

*Keywords:*
Adaptive dynamic programming
Convergence analysis
Heuristic dynamic programming
Neural networks
Optimal tracking control
Reinforcement learning

ABSTRACT

In this paper, an optimal tracking control scheme is proposed for a class of unknown discrete-time nonlinear systems using iterative adaptive dynamic programming (ADP) algorithm. First, in order to obtain the dynamics of the system, an identifier is constructed by a three-layer feedforward neural network (NN). Second, a feedforward neuro-controller is designed to get the desired control input of the system. Third, via system transformation, the original tracking problem is transformed into a regulation problem with respect to the state tracking error. Then, the iterative ADP algorithm based on heuristic dynamic programming is introduced to deal with the regulation problem with convergence analysis. In this scheme, feedforward NNs are used as parametric structures for facilitating the implementation of the iterative algorithm. Finally, simulation results are also presented to demonstrate the effectiveness of the proposed scheme.



## 1. Introduction

As is well known, there are demands for optimal tracking controller designs in many real world applications. For solving the optimal tracking control problems, many approaches were proposed in the last several decades [1–6]. However, these traditional control approaches, such as feedback linearization [1] and plant inversion [2], usually involve complex mathematical analysis and yet have many difficulties in controlling highly nonlinear plants. For feedback linearization, the obtained control is only effective in the neighborhood of the equilibrium point. For plant inversion, the exact inversion model of the plant is quite difficult to obtain, if not impossible. Moreover, for unknown nonlinear systems, it is more difficult to get the optimal tracking control by traditional approaches. Therefore, it is necessary to design new approaches and direct optimal control schemes for unknown nonlinear systems. When dealing with the optimal tracking problems, the difficulty is the requirement of solving the nonlinear Hamilton–Jacobi–Bellman (HJB) equation which is usually too difficult to solve analytically. Though it is well known that dynamic programming (DP) is an useful approach for solving optimal control problems, it is often computationally untenable to run DP due to the backward numerical process required for its solutions, i.e., due to the "curse of dimensionality" [7,8].

In recent years, in order to obtain approximate solutions of the HJB equation, adaptive dynamic programming (ADP) algorithms have attracted much attention from researchers [9–39]. ADP was proposed in [10–13] as a way for solving optimal control problems forward-in-time. There are several synonyms used for ADP including "adaptive dynamic programming" [14,15], "approximate dynamic programming" [11], "neuro-dynamic programming" [16] and "adaptive critic designs" [17]. In [18], ADP approaches were classified into several main schemes including heuristic dynamic programming (HDP), action-dependent heuristic dynamic programming (ADHDP), dual heuristic dynamic programming (DHP), ADDHP, globalized DHP (GDHP) and ADGDHP. Al-Tamimi et al. [19] proposed a greedy iterative HDP to solve the optimal control problem for nonlinear discrete-time systems. Vrabie et al. [20] studied the continuous-time optimal control problem using ADP. Wang et al. [21] developed an $\varepsilon$-ADP algorithm for studying finite-horizon optimal control of discrete-time nonlinear systems. Dierks et al. [22] relaxed the requirement of explicit knowing the dynamics of system via two processes: online identification of the system and offline training of the optimal controller. Park et al. [23] used multilayer neural networks (NNs) to design a finite-horizon optimal tracking neuro-controller for discrete-time nonlinear systems with quadratic cost function. Dierks et al. [24] used the ADP technique to solve the optimal tracking control of affine nonlinear discrete-time systems with partially unknown nonlinear dynamics. Zhang et al. [25] gave a novel infinite-horizon optimal tracking control scheme based on the greedy HDP algorithm for discrete-time nonlinear systems with the requirement of fully known system dynamics.

* Corresponding author. Tel.: +86 10 62557379; fax: +86 10 62650912.
*E-mail addresses:* yuzhu.huang@ia.ac.cn (Y. Huang), derong.liu@ia.ac.cn, derongliu@gmail.com (D. Liu).

0925-2312/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.neucom.2012.07.047

However, to the best of our knowledge, there is still no result for solving the optimal tracking control problems for completely unknown discrete-time nonlinear systems via ADP technique. In this paper, it is the first time to use the iterative ADP algorithm to design an infinite-time optimal tracking controller for a class of discrete-time nonlinear systems with unknown nonlinear dynamics. For designing the optimal tracking controller, it is necessary to set up two parametric structures, namely, the NN identifier and the feedforward neuro-controller. The NN identifier is to learn the dynamics of the unknown system, and the feedforward neuro-controller is designed to obtain the desired control for keeping the system state to a given reference value. Then, after transforming the tracking problem to a regulation problem, the optimal feedback control is obtained by the iterative ADP algorithm.

The rest of this paper is organized as follows. In Section 2, we present the problem statement, and transform the tracking control problem into a regulation problem. In Section 3, we demonstrate the establishments of the NN identifier and feedforward neuro-controller with convergence analysis. In Section 4, we present the derivation of the iterative ADP algorithm with convergence analysis, and then we illustrate the NN implementation of the iterative ADP algorithm for solving the optimal tracking problem. In Section 5, we present simulation results to demonstrate the effectiveness of the proposed optimal tracking control scheme. In Section 6, we conclude the paper with a few remarks.

## 2. Problem statement

Consider the nonlinear discrete-time system given by

$$\begin{aligned} x(k+1) &= F(x(k),u(x(k))) \\ &= f(x(k))+g(x(k))u(x(k)), \end{aligned} \tag{1}$$

where $x(k)\in\mathbb{R}^n$ represents the state vector of the system, $u(x(k))\in\mathbb{R}^m$ represents the control vector, $f(\cdot)$ and $g(\cdot)$ are differentiable in their argument with $f(0)=0$. Assume that $f+gu$ is Lipschitz continuous on a set $\Omega$ in $\mathbb{R}^n$ containing the origin, and $g(x(k))$ satisfies $\|g(x(k))\|_F\le g_M$ [9]. Besides, it is assumed that there exists a matrix function $g^{-1}(x(k))\in\mathbb{R}^{m\times n}$ such that $g^{-1}(x(k))g(x(k))=I\in\mathbb{R}^{m\times m}$ where $I$ is the identity matrix. Without loss of generality, assume that the system is controllable in the sense that there exists a continuous control on $\Omega$ that asymptotically stabilizes the system.

For the optimal tracking control problem, the control objective is to find the optimal control $u^*(x(k))$, so as to make the nonlinear system (1) to track a reference (desired) trajectory $r(k)$ in an optimal manner. We assumed that the reference trajectory $r(k)$ satisfies

$$r(k+1)=\varphi(r(k)), \tag{2}$$

where $r(k)\in\mathbb{R}^n$ and $\varphi(r(k))\in\mathbb{R}^n$, and it is assumed that the mapping between the state $x(k)$ and the desired trajectory $r(k)$ is one-to-one. For convenience, in the sequel, $u(x(k))$ is denoted as $u(k)$.

The tracking error can be defined as follows:

$$e(k)=x(k)-r(k). \tag{3}$$

Assume that the desired trajectory $r(k)$ satisfies the following form:

$$r(k+1)=f(r(k))+g(r(k))u_d(k), \tag{4}$$

where $r(k+1)$ is the reference output and $u_d(k)$ denotes the desired control input.

If the dynamics of the system is known and the inverse of the control coefficient matrix function $g^{-1}(r(k))$ exists, the desired control $u_d(k)$ can be obtained by

$$u_d(k)=g^{-1}(r(k))(\varphi(r(k))-f(r(k))), \tag{5}$$

where $g^{-1}(r(k))g(r(k))=I$ and $I\in\mathbb{R}^{m\times m}$ is the identity matrix. Then, the feedback control $u_e(k)$ is defined by

$$u_e(k)=u(k)-u_d(k). \tag{6}$$

Considering (1)–(6), the tracking error $e(k+1)$ can be expressed as

$$\begin{aligned} e(k+1) &= x(k+1)-r(k+1) \\ &= f(e(k)+r(k))+g(e(k)+r(k))u_e(k) \\ &\quad +g(e(k)+r(k))g^{-1}(r(k))(\varphi(r(k)) \\ &\quad -f(r(k)))-\varphi(r(k)). \end{aligned} \tag{7}$$

From (7), we can see that if the reference trajectory $r(k)$ is given, we can use $e(k)$ and $u_e(k)$ to obtain the tracking error $e(k+1)$. For convenience of analysis, let (7) be rewritten as

$$e(k+1)=f_e(k)+g_e(k)u_e(k), \tag{8}$$

where
$f_e(k)=g(e(k)+r(k))g^{-1}(r(k))(\varphi(r(k))-f(r(k)))+f(e(k)+r(k))-\varphi(r(k))$
and $g_e(k)=g(e(k)+r(k))$. For infinite horizon optimal tracking control problem, it is desired to find the optimal control law $v(e)$ so that the control sequence $u_e(\cdot)=(u_e(k),u_e(k+1),\ldots)$ with each $u_e(i)=v(e(i))\in\mathbb{R}^m$, $i=k,\ k+1,\ldots$, minimizes the following cost function:

$$J(e(k),u_e(\cdot))=\sum_{i=k}^{\infty}U(e(i),\ u_e(i)), \tag{9}$$

where $U$ is the utility function, $U(0,0)=0$, $U(e(i),\ u_e(i))\ge 0$ for $\forall e(i),\ u_e(i),\ i=k,\ k+1,\ldots$. In this paper, the utility function $U$ is chosen as the quadratic form

$$U(e(i),\ u_e(i))=e^T(i)Qe(i)+u_e^T(i)Ru_e(i), \tag{10}$$

where $Q\in\mathbb{R}^{n\times n}$ and $R\in\mathbb{R}^{m\times m}$ are symmetric and positive definite. This quadratic cost function will not only force the system state to follow the reference trajectory, but also force the system input to be close to the desired control in maintaining the state to its reference value. Thus, for the optimal control problem, the state feedback control law $v(e)$ must not only stabilize the system (8) on $\Omega$ but also guarantee that (9) is finite, i.e., the control law must be admissible.

**Definition 1.** A control law $v(e)$ is said to be admissible with respect to (9) on $\Omega$, if $v(e)$ is continuous on a compact set $\Omega_u\in\mathbb{R}^m$ for $\forall e(k)\in\Omega$, and stabilizes (8) on $\Omega$, $v(0)=0$, and for $e(0)\in\Omega$, $J(e(0),\ u_e(\cdot))$ is finite, where $u_e(\cdot)=(u_e(0),u_e(1),\ldots)$ and $u_e(k)=v(e(k))$, $k=0,1,\ldots$.

For infinite horizon optimal control problem, based on the above definition, a control law sequence $\eta_i=(\eta_\infty,\ldots,\eta_1,\eta_0)$ is called admissible if the resulting control sequence $(u_e(0),u_e(1),\ldots,u_e(\infty))$ stabilizes (8) with any initial state $e(0)$ and guarantees that $J(e(0),u_e(\cdot))$ is finite. The corresponding final state is denoted as $e^f(e(k),u_e^{k\to\infty})=\lim_{k\to\infty}e(k)$, where
$u_e^{k\to\infty}=(u_e(k),u_e(k+1),\ldots,u_e(\infty))$.

Let

$$\Theta=\{u_e^{k\to\infty}:e^f(e(k),u_e^{k\to\infty})=0\} \tag{11}$$

be the set of all infinite horizon admissible control sequence of $e(k)$. Define the optimal cost function $J^*(e(k))$ as

$$J^*(e(k))=\inf_{u_e^{k\to\infty}}\{J(e(k),u_e^{k\to\infty}):u_e^{k\to\infty}\in\Theta\}, \tag{12}$$

and the corresponding optimal feedback control is $u_e^*(k)$ which minimizes (9).

In addition, considering (5) and (6), we have

$$u(k)=u_e(k)+u_d(k). \tag{13}$$

It is observed that the tracking control $u(k)$ consists of a feedback control $u_e(k)$ and a predetermined desired control $u_d(k)$ corresponding to the reference trajectory $r(k)$. Moreover, we should note that as long as the proble m is limited to the case where $u_d(k)$ exists for any $r(k)$, it is reasonable to assume that the optimal tracking control has the above form.

Before proceeding, the following technical lemma is needed.

**Lemma 1.** *Let $u_e(k)$ be an admissible control such that the tracking error system (8) is asymptotically stable. Then, the internal dynamics $f_e(k)$ is bounded satisfying*

$$\|f_e(k)\|^2 \le \Upsilon\lambda_{\min}(Q)\|e(k)\|^2/2+(\Upsilon\lambda_{\min}(R)-2g_M^2)\|u_e(k)\|^2/2, \tag{14}$$

*where $\lambda_{\min}(R)$ is the minimum eigenvalue of $R$, $\lambda_{\min}(Q)$ is the minimum eigenvalue of $Q$, and $\Upsilon > 2g_M^2/\lambda_{\min}(R)$ is a known positive constant.*

**Proof.** Considering the following positive definite Lyapunov function candidate:

$$V(k) = e^T(k)e(k)+\Upsilon J_e(k), \tag{15}$$

where $J_e(k) = J(e(k), u_e(k))$ is defined in (9). The first difference of the Lyapunov function candidate is given by

$$\Delta V(k) = e^T(k+1)e(k+1)-e^T(k)e(k)+\Upsilon\Delta J_e(k). \tag{16}$$

Using (8) and (9), we have

$$\begin{aligned}\Delta V(k) = {} & (f_e(k)+g_e(k)u_e(k))^T(f_e(k)+g_e(k)u_e(k))\\ & -e^T(k)e(k)-\Upsilon e^T(k)Qe(k)+u_e^T(k)Ru_e(k)).\end{aligned} \tag{17}$$

After applying the Cauchy–Schwarz inequality, we can obtain

$$\begin{aligned}\Delta V(k) \le {} & 2\|f_e(k)\|^2-(\Upsilon\lambda_{\min}(R)-2g_M^2)\|u_e(k)\|^2\\ & -\Upsilon\lambda_{\min}(Q)\|e(k)\|^2-\|e(k)\|^2.\end{aligned} \tag{18}$$

Considering the goal of the tracking error system (8) being asymptotically stable, i.e., $\Delta V(k) < 0$, we require

$$\|f_e(k)\|^2 \le \Upsilon\lambda_{\min}(Q)\|e(k)\|^2/2+(\Upsilon\lambda_{\min}(R)-2g_M^2)\|u_e(k)\|^2/2. \tag{19}$$

Therefore, if the bound in (19) is true, we can get $\Delta V(k) < 0$, implying the asymptotic stability of (8). □

**Remark 1.** Lemma 1 shows that if the internal dynamics $f_e(k)$ is bounded satisfying (14), then, for the nonlinear system (8), there exists an admissible control $u_e(k)$ not only stabilizes the system (8) on $\Omega$ but also guarantees that (9) is finite.

Assume that the system dynamics are known, we can design the controller by the aforementioned method. However, for most cases, obtaining the complete, even partial knowledge of the nonlinear system dynamics is a challenging task. For unknown nonlinear systems, an NN identifier is constructed to learn the system dynamics, and a feedforward neuro-controller is designed using NNs with the aid of the NN identifier to obtain the desired control $u_d(k)$. Furthermore, according to (13), for obtaining the optimal tracking control, it is necessary to find the optimal feedback control $u_e^*(k)$ for the system (8) with respect to (9). In this sense it can be said that the optimal tracking problem of (1) is transformed into an optimal regulation problem of (8). In the following, we will focus on how to get the optimal feedback control.

## 3. Establishments of NN identifier and feedforward neuro-controller

In this section, a three-layer feedforward NN identifier is established to reconstruct the unknown system dynamics using available input–output data. Let the number of hidden layer neurons be denoted as $d$, the ideal weights between the input layer and the hidden layer be denoted as $v_m^*$, the ideal weights between the hidden layer and the output layer be denotes as $\omega_m^*$. According to the universal approximation property of NNs, the system dynamics (1) has an NN representation on a compact set $S$, which can be written as

$$x(k+1) = \omega_m^{*T}\sigma(v_m^{*T}z(k))+\varepsilon_m(k), \tag{20}$$

where $\omega_m^* \in \mathbb{R}^{d\times n}$ and $v_m^* \in \mathbb{R}^{(n+m)\times d}$ are the constant ideal weight matrices, $z(k) = [x^T(k)\ u^T(k)]^T$ is the NN input, and $\varepsilon_m(k)$ is the bounded NN functional approximation error, $\sigma(\cdot)$ is the NN activation function and selected to be hyperbolic tangent function. Besides, the NN activation functions are bounded such that $\|\sigma(\cdot)\| \le \sigma_M$ for a constant $\sigma_M$.

For convenience of analysis, only the output weights are updated during the training, while the hidden weights are kept fixed. So the activation function $\sigma(v_m^{*T}z(k))$ can be rewritten as $\sigma(\bar{z}(k))$ in the hidden layer, where $\bar{z}(k) = v_m^{*T}z(k), \bar{z}(k) \in \mathbb{R}^l$. Thus, the NN model for the system is constructed as

$$\hat{x}(k+1) = \hat{\omega}_m^T(k)\sigma(\bar{z}(k)), \tag{21}$$

where $\hat{x}(k)$ is the estimated system state vector and $\hat{\omega}_m(k)$ is the estimation of the ideal weight matrix $\omega_m^*$.

Then, we define the system identification error as

$$\begin{aligned}\tilde{x}(k+1) &= \hat{x}(k+1)-x(k+1)\\ &= \tilde{\omega}_m(k)\sigma(\bar{z}(k))-\varepsilon_m(k),\end{aligned} \tag{22}$$

where $\tilde{\omega}_m(k) = \hat{\omega}_m(k)-\omega_m^*$. Let $\phi(k) = \tilde{\omega}_m^T(k)\sigma(\bar{z}(k))$. Thus, we have

$$\tilde{x}(k+1) = \phi(k)-\varepsilon_m(k). \tag{23}$$

The weights are adjusted to minimize the following error

$$E(k+1) = \tfrac{1}{2}\tilde{x}^T(k+1)\tilde{x}(k+1). \tag{24}$$

Using the gradient-based adaptation rule, the weights are updated by

$$\begin{aligned}\hat{\omega}_m(k+1) &= \hat{\omega}_m(k)-l_m\frac{\partial E(k+1)}{\tilde{x}(k+1)}\frac{\partial\tilde{x}(k+1)}{\tilde{w}_m(k)}\\ &= \hat{\omega}_m(k)-l_m\sigma(\bar{z}(k))\tilde{x}^T(k+1),\end{aligned} \tag{25}$$

where $l_m > 0$ is the learning rate.

Inspired by the work in [22], before proceeding, we need to give the following assumption which is considered mild in comparison with the approximation error being bounded by a known constant.

**Assumption 1.** The NN approximation error $\varepsilon_m(k)$ is assumed to be upper bounded by a function of estimation error such that

$$\varepsilon_m^T(k)\varepsilon_m(k) \le \lambda_M\tilde{x}^T(k)\tilde{x}(k), \tag{26}$$

where $\lambda_M$ is a finite constant value.

Next, the stability analysis of the system identification scheme is presented by using the Lyapunov method.

**Theorem 1.** *Let the identification scheme (21) be used to identify the nonlinear system (1), and let the NN weights be updated by (25), then, the system identification error $\tilde{x}(k)$ is asymptotically stable while the parameter estimation error $\tilde{\omega}_m(k)$ is bounded.*

**Proof.** Considering the following positive definite Lyapunov function candidate:

$$J(k) = \tilde{x}^T(k)\tilde{x}(k)+\frac{1}{l_m}\mathrm{tr}\{\tilde{\omega}_m^T(k)\tilde{\omega}_m(k)\}. \tag{27}$$

The first difference of the Lyapunov function candidate is given by

$$\begin{aligned}\Delta J(k) = {} & \tilde{x}^T(k+1)\tilde{x}(k+1)-\tilde{x}^T(k)\tilde{x}(k)\\ & +\frac{1}{l_m}\mathrm{tr}\{\tilde{\omega}_m^T(k+1)\tilde{\omega}_m(k+1)-\tilde{\omega}_m^T(k)\tilde{\omega}_m(k)\}.\end{aligned} \tag{28}$$

With the identification error dynamics (23) and the weight tuning rule (25), we can obtain

$$\begin{aligned}\Delta J(k) &= \phi^T(k)\phi(k) - 2\phi^T(k)\varepsilon_m(k) + \varepsilon_m^T(k)\varepsilon_m(k) \\ &\quad + l_m\sigma^T(\bar{z}(k))\sigma(\bar{z}(k))\tilde{x}^T(k+1)\tilde{x}(k+1) \\ &\quad - \tilde{x}^T(k)\tilde{x}(k) - 2\phi^T(k)\tilde{x}(k+1). \end{aligned} \tag{29}$$

Applying the Cauchy–Schwarz inequality, we can get

$$\begin{aligned}\Delta J(k) &\le -\phi^T(k)\phi(k) + \varepsilon_m^T(k)\varepsilon_m(k) - \tilde{x}^T(k)\tilde{x}(k) \\ &\quad + 2l_m\sigma^T(\bar{z}(k))\sigma(\bar{z}(k))(\phi^T(k)\phi(k) + \varepsilon_m^T(k)\varepsilon_m(k)). \end{aligned} \tag{30}$$

Considering $\|\sigma(\bar{z}(k))\| \le \sigma_M$ and (26), we have

$$\begin{aligned}\Delta J(k) &\le -(1-2l_m\sigma_M^2)\|\phi(k)\|^2 \\ &\quad -(1-\lambda_M-2l_m\lambda_M\sigma_M^2)\|\tilde{x}(k)\|^2. \end{aligned} \tag{31}$$

Let $l_m$ be selected as $l_m \le \beta^2/2\sigma_M^2$. Then, (31) becomes

$$\Delta J(k) \le -(1-\beta^2)\|\phi(k)\|^2 - (1-\lambda_M-\lambda_M\beta^2)\|\tilde{x}(k)\|^2. \tag{32}$$

Therefore, $\Delta J(k) \le 0$ provided that $0 \le \lambda_M \le 1$ and

$$\max\left\{-\sqrt{\frac{1-\lambda_M}{\lambda_M}}, -1\right\} \le \beta \le \min\left\{\sqrt{\frac{1-\lambda_M}{\lambda_M}}, 1\right\},$$

where $\beta \ne 0$. As long as the parameters are selected as discussed above, we have $\Delta J(k) \le 0$. Therefore, the identification error $\tilde{x}(k)$ and the weight estimation error $\tilde{\omega}_m(k)$ are bounded provided $\tilde{x}(0)$ and $\tilde{\omega}_m(0)$ are bounded in the compact set $S$. Furthermore, according to [9], by summing both side of (32) to infinity and taking limits when $k \to \infty$, it can be concluded that the estimation error $\tilde{x}(k)$ approaches zero with $k \to \infty$. □

After a sufficiently long learning process, the NN identification error approaches zero, and the system dynamics (1) can be written as

$$x(k+1) = \hat{x}(k+1) = \hat{\omega}_m^T(k)\sigma(\bar{z}(k)). \tag{33}$$

In the following, optimal tracking control for the unknown nonlinear system will be achieved using the result of the system identification in (33). In order to obtain the desired control $u_d(k)$, a feedforward neuro-controller is constructed to learn the inverse dynamics of the system by a three-layer feedforward NN.

Note that the desired control can be obtained by setting $x(k) = r(k)$ and $u(k) = u_d(k)$ for all $k$ in the original system (1), i.e.,

$$r(k+1) = F(r(k), u_d(k))$$

$$u_d(k) = F^{-1}(r(k+1), r(k)) \tag{34}$$

where $u_d(k)$ is the desired control and $F^{-1}$ is the inverse function of $F$. For the unknown nonlinear system, since it is almost impossible to get the specific solution of $F^{-1}$, we introduce the feedforward neuro-controller as an inverse mapping of the system. The feedforward neuro-controller is established with the aid of the trained NN identifier as shown in Fig. 1.

The feedforward neuro-controller is trained to adjust its weight parameters so that the output of the NN identifier $\hat{r}(k+1)$ approximates the given reference value $r(k+1)$. It is clear that the feedforward neuro-controller is trained for driving the output of the NN identifier to approximate the reference trajectory. Thus, we can consider that the purpose of training the feedforward neuro-controller is to learn the inverse dynamics of the system for generating the desired control $u_d(k)$.

From Fig. 1, the feedforward neuro-controller outputs $\hat{u}_d(k)$. Then, $\hat{u}_d(k)$ is propagated to the NN identifier for outputing the estimated reference trajectory $\hat{r}(k+1)$. Thus, it is reasonable to consider that the objective of training the feedforward neuro-controller is to minimize the error function defined by

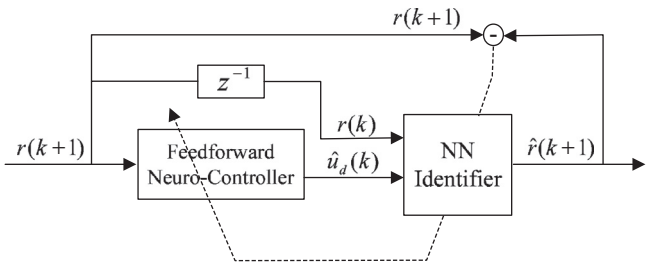


**Fig. 1.** The structure diagram of the feedforward neuro-controller.

$$E_g(k+1) = \tfrac{1}{2}e_g^T(k+1)e_g(k+1), \tag{35}$$

where $e_g(k+1) = \hat{r}(k+1) - r(k+1)$. In addition, it should be noted that the error is propagated backward through the NN identifier to adjust the weight parameters in the feedforward neuro-controller. The weight update rule for the feedforward neuro-controller is the gradient-based adaptation given by

$$\omega_g(k+1) = \omega_g(k) + \Delta\omega_g(k), \tag{36}$$

$$\Delta\omega_g(k) = -l_g\frac{\partial E_g(k+1)}{\partial\omega_g(k)}, \tag{37}$$

$$\frac{\partial E_g(k+1)}{\partial\omega_g(k)} = \frac{\partial E_g(k+1)}{\partial\hat{r}(k+1)}\frac{\partial\hat{r}(k+1)}{\partial\hat{u}_d(k)}\frac{\partial\hat{u}_d(k)}{\partial\omega_g(k)}, \tag{38}$$

where $l_g > 0$ is the learning rate and $\omega_g$ is the weight vector in the feedforward neuro-controller. Note that $\partial\hat{r}(k+1)/\partial\hat{u}_d(k)$ can be obtained by backpropagation from the output of the NN identifier $\hat{r}(k+1)$ to its input $\hat{u}_d(k)$.

Before starting the training, some small random values are selected to initialize the weight parameters in feedforward neuro-controller. At the end of training, the weight parameters in the feedforward neuro-controller are adjusted so that the output of the NN identifier $\hat{r}(k+1)$ can follow the reference value $r(k+1)$. Hence, we can consider that the feedforward neuro-controller has learned the steady-state inverse dynamics of the system with the aid of the NN identifier.

**Remark 2.** Considering (34), it should be noted that the inverse is not unique in general and any solution is sufficient for control purpose, i.e., $\hat{u}_d(k)$ drives the output of the NN identifier $\hat{r}(k+1)$ to approximate the reference value $r(k+1)$. However, considering the control input is control energy, the smallest solution of $\hat{u}_d(k)$ is preferred. Therefore, the training of feedforward neuro-controller begins with small random values of weight parameters. This allows the desired control to grow from a small random value and converge to the smallest solution of $\hat{u}_d(k)$, which is preferred over all other possible solutions. Similar results can be seen in [23].

According to (8) and (9) discussed previously, the optimal tracking problem of (1) has been transformed into an optimal regulation problem. Therefore, in the following, a greedy iterative ADP algorithm is introduced to obtain the optimal feedback control which will not only stabilize the system but also guarantee the cost function to be finite.

## 4. The iterative ADP algorithm

Three parts are included in this section. The derivation of the iterative ADP algorithm is provided in the first part, the corresponding convergence proof is presented in the second part, and

the implementation of the optimal control scheme is described in the third part.

### 4.1. The derivation of the iterative ADP algorithm

In this part, we present the iterative ADP algorithm based on Bellman's principle of optimality and the greedy iterative principle.

First, we start with the initial cost function $V_0(\cdot)=0$ which is not necessarily the optimal value function, and then the law of single control vector $v_0(e(k))$ can be solved by

$$v_0(e(k))=\arg\min_{u_e(k)}\{e^T(k)Qe(k)+u_e^T(k)Ru_e(k)+V_0(e(k+1))\}. \tag{39}$$

Then, we update the cost function as

$$\begin{aligned}V_1(e(k))&=\min_{u_e(k)}\{e^T(k)Qe(k)+u_e^T(k)Ru_e(k)+V_0(e(k+1))\}\\&=e^T(k)Qe(k)+v_0^T(e(k))Rv_0(e(k)).\end{aligned} \tag{40}$$

Therefore, for $i=1,2,\ldots$, the iterative ADP algorithm can be implemented by the iterations between

$$\begin{aligned}v_i(e(k))&=\arg\min_{u_e(k)}\{e^T(k)Qe(k)+u_e^T(k)Ru_e(k)+V_i(e(k+1))\}\\&=-\frac{1}{2}R^{-1}g^T(e(k)+r(k))\frac{\partial V_i(e(k+1))}{\partial e(k+1)}\end{aligned} \tag{41}$$

and

$$\begin{aligned}V_{i+1}(e(k))&=\min_{u_e(k)}\{e^T(k)Qe(k)+u_e^T(k)Ru_e(k)+V_i(e(k+1))\}\\&=e^T(k)Qe(k)+v_i^T(e(k))Rv_i(e(k))+V_i(e(k+1))\end{aligned} \tag{42}$$

where $e(k+1)=F(e(k),v_i(e(k)))$, $i$ represents the iterative index of the control law and the cost function, while $k$ represents the time index of the system state trajectory. In addition, in the iterative ADP algorithm, it is worth noting that the cost function and the control law are updated by recursive iterations with the iterative index $i$ increasing from 0 and the initial cost function $V_0(\cdot)=0$.

### 4.2. Convergence analysis of the iterative ADP algorithm

Next, we present a convergence proof of the iteration between (41) and (42), with the cost function $V_i\to J^*$ and the control law $v_i\to u_e^*$ as $i\to\infty$.

**Lemma 2.** *Let $\{\mu_i\}$ be an arbitrary sequence of control laws and $\{v_i\}$ be the control sequence expressed in (41). Define $V_i(e(k))$ as in (42) and $\Lambda_i$ as*

$$\Lambda_{i+1}(e(k))=e^T(k)Qe(k)+\mu_i^T(e(k))R\mu_i(e(k))+\Lambda_i(e(k+1)). \tag{43}$$

*If $V_0(e(k))=\Lambda_0(e(k))=0$, then $V_i(e(k))\le\Lambda_i(e(k))$, $\forall i$.*

**Proof.** It can clearly be seen that $V_{i+1}(e(k))$ is the result of minimizing the right-hand side of (42) with respect to the control input $u_e(k)$, while $\Lambda_{i+1}(e(k))$ is a result of arbitrary control input. □

**Lemma 3.** *Let the cost function sequence $\{V_i\}$ be defined in (42). If the system is controllable, then there exists an upper bound $Y$ such that $0\le V_i(e(k))\le Y$, $\forall i$.*

**Proof.** Let $\{\eta_i(e)\}$ be a sequence of admissible control laws, and let $Z_0(\cdot)=V_0(\cdot)=0$, where $V_i(e(k))$ is updated by (42) and $Z_i$ is updated by

$$Z_{i+1}(e(k))=e^T(k)Qe(k)+\eta_i^T(e(k))R\eta_i(e(k))+Z_i(e(k+1)). \tag{44}$$

Considering (44), we can further have

$$\begin{aligned}Z_i(e(k+1))&=e^T(k+1)Qe(k+1)+\eta_{i-1}^T(e(k+1))R\eta_{i-1}(e(k+1))\\&\quad+Z_{i-1}(e(k+2)).\end{aligned} \tag{45}$$

With $U(e(k),\eta_i(e(k)))=e^T(k)Qe(k)+\eta_i^T(e(k))R\eta_i(e(k))$, the following relation can be obtained:

$$\begin{aligned}Z_{i+1}(e(k))&=U(e(k),\eta_i(e(k)))+U(e(k+1),\eta_{i-1}(e(k+1)))+Z_{i-1}(e(k+2))\\&=U(e(k),\eta_i(e(k)))+U(e(k+1),\eta_{i-1}(e(k+1)))\\&\quad+U(e(k+2),\eta_{i-2}(e(k+2)))+Z_{i-2}(e(k+3))\\&\quad\vdots\\&=U(e(k),\eta_i(e(k)))+U(e(k+1),\eta_{i-1}(e(k+1)))\\&\quad+U(e(k+2),\eta_{i-2}(e(k+2)))\\&\quad+\cdots+U(e(k+i),\eta_0(e(k+i)))+Z_0(e(k+i+1)),\end{aligned} \tag{46}$$

where $Z_0(e(k+i+1))=0$. Then, (46) can be written as

$$Z_{i+1}(e(k))=\sum_{j=0}^{i}U(e(k+j),\eta_{i-j}(e(k+j))). \tag{47}$$

Considering the fact that $U$ is positive semidefinite, we have

$$Z_{i+1}(e(k))\le\lim_{i\to\infty}\sum_{j=0}^{i}U(e(k+j),\eta_{i-j}(e(k+j))). \tag{48}$$

Note that $\{\eta_i(e)\}$ is an admissible control law sequence, i.e., $e(k)\to0$ as $k\to\infty$. Thus, for $\forall i$, there exists an upper bounded $Y$ such that

$$Z_{i+1}(e(k))\le\lim_{i\to\infty}\sum_{j=0}^{i}U(e(k+j),\eta_{i-j}(e(k+j)))\le Y. \tag{49}$$

Then, combining with Lemma 2, we can get

$$V_{i+1}(e(k))\le Z_{i+1}(e(k))\le Y\quad\forall i. \qquad\square \tag{50}$$

Next, based on Lemmas 2 and 3, we present our main theorems.

**Theorem 2.** *Let the control law sequence $\{v_i\}$ and the cost function sequence $\{V_i\}$ be defined in (41) and (42), respectively, with $V_0(\cdot)=0$. Then, $V_i(e(k))$ is a nondecreasing sequence satisfying $V_i(e(k))\le V_{i+1}(e(k))$, $\forall i$. Moreover, as $i\to\infty$, $V_i\to J^*$, i.e.,*

$$\lim_{i\to\infty}V_i(e(k))=J^*(e(k)). \tag{51}$$

**Proof.** Define a new sequence $\{\Phi_i\}$ as follows:

$$\Phi_{i+1}(e(k))=e^T(k)Qe(k)+v_{i+1}^T(e(k))Rv_{i+1}(e(k))+\Phi_i(e(k+1)) \tag{52}$$

with $\Phi_0(\cdot)=V_0(\cdot)=0$. The control law sequence $\{v_i\}$ and the cost function sequence $\{V_i\}$ are updated by (41) and (42), respectively.

In the following, we prove $\Phi_i(e(k))\le V_{i+1}(e(k))$ by mathematical induction.

First, we prove that it holds for $i=0$. Considering

$$V_1(e(k))-\Phi_0(e(k))=e^T(k)Qe(k)+v_0(e(k))Rv_0(e(k))\ge0, \tag{53}$$

thus, for $i=0$, we have

$$\Phi_0(e(k))\le V_1(e(k)). \tag{54}$$

Second, it is assumed that it holds for $i-1$, i.e., $\Phi_{i-1}(e(k))\le V_i(e(k))$, $\forall e(k)$. For $i$, since

$$\Phi_i(e(k))=e^T(k)Qe(k)+v_i^T(e(k))Rv_i(e(k))+\Phi_{i-1}(e(k+1)) \tag{55}$$

and

$$V_{i+1}(e(k))=e^T(k)Qe(k)+v_i^T(e(k))Rv_i(e(k))+V_i(e(k+1)) \tag{56}$$

hold, we can get

$$V_{i+1}(e(k))-\Phi_i(e(k))=V_i(e(k+1))-\Phi_{i-1}(e(k+1))\ge0,$$

i.e.,

$$\Phi_i(e(k)) \le V_{i+1}(e(k)). \tag{57}$$

Therefore, (57) is true for any $i$ by mathematical induction.

Furthermore, from Lemma 2, we have $V_i(e(k)) \le \Phi_i(e(k))$. Therefore, we can get

$$V_i(e(k)) \le \Phi_i(e(k)) \le V_{i+1}(e(k)). \tag{58}$$

Moving on, let $\{\eta_i^{(l)}\}$ be the $l$th admissible control law sequence. We define the associated sequence $P_i^{(l)}(e(k))$ as follows:

$$P_{i+1}^{(l)}(e(k)) = \sum_{j=0}^{i} U(e(k+j),\eta_{i-j}^{(l)}(e(k))). \tag{59}$$

Let $i \to \infty$, we have

$$P_{\infty}^{(l)}(e(k)) = \lim_{i\to\infty} \sum_{j=o}^{i} U(e(k+j),\eta_{i-j}^{(l)}(e(k))). \tag{60}$$

Note that the control law sequence $\{\eta_i^{(l)}\}$ is admissible. Thus, $P_{\infty}^{(l)}(e(k))$ is finite. For any $l$, there exists an upper bound $Y_l$ such that

$$V_{i+1}(e(k)) \le P_{i+1}^{(l)}(e(k)) \le Y_l, \quad \forall l,i. \tag{61}$$

According to the definition of $J^*(e(k))$ in (12), for any $\varepsilon > 0$, there exists a sequence of admissible control law $\{\eta_i^{(l)}\}$ such that the corresponding cost function satisfies $P_{\infty}^{(l)}(e(k)) \le J^*(e(k)) + \varepsilon$. With (61), we can obtain

$$\lim_{i\to\infty} V_i(e(k)) \le P_{\infty}^{(l)}(e(k)) \le J^*(e(k)) + \varepsilon. \tag{62}$$

Since $\varepsilon$ is chosen arbitrarily, we have

$$\lim_{i\to\infty} V_i(e(k)) \le J^*(e(k)). \tag{63}$$

On the other hand, since $V_{i+1}(e(k)) \le P_{i+1}^{(l)}(e(k)) \le Y_l, \forall l,i$, we have $\lim_{i\to\infty} V_i(e(k)) \le \inf_l\{Y_l\}$. According to the definition of admissible control law sequence, the control law sequence associated with the cost function $\lim_{i\to\infty} V_i(e(k))$ must be an admissible control law sequence. Thus, there exists a sequence of admissible control laws $\{\eta_i^{(L)}\}$ which satisfies $\lim_{i\to\infty} V_i(e(k)) = P_{\infty}^{(L)}(e(k))$. Since $J^*(e(k))$ is the infimum cost function of all admissible control sequences starting from $e(k)$ with length $\infty$, we have

$$\lim_{i\to\infty} V_i(e(k)) \ge J^*(e(k)). \tag{64}$$

Considering (63) and (64), we have $\lim_{i\to\infty} V_i(e(k)) = J^*(e(k))$, i.e., $J^*(e(k))$ is the limit of the cost function sequence $\{V_i(e(k))\}$. □

In the above, we have completed the proof of Theorem 2. Hence, we can conclude that the cost function sequence $\{V_i\}$ is a monotonically nondecreasing sequence with an upper bound and $J^*$ is the limit of the cost function sequence $\{V_i\}$. Next, when we make $i \to \infty$ in (42), we can get the following result.

**Theorem 3.** *For any $e(k)$, define*

$$\lim_{i\to\infty} V_i(e(k)) = V_{\infty}(e(k))$$

*as the limit of the cost function sequence $\{V_i\}$. $V_{\infty}(e(k))$ satisfies the HJB equation*

$$V_{\infty}(e(k)) = \min_{u_e(k)}\{e^T(k)Qe(k) + u_e^T(k)Ru_e(k) + V_{\infty}(e(k+1))\}. \tag{65}$$

**Proof.** For any $i$ and $u_e(k)$, according to (42), we have

$$V_i(e(k)) \le e^T(k)Qe(k) + u_e^T(k)Ru_e(k) + V_{i-1}(e(k+1)). \tag{66}$$

Then, considering Theorem 2, we have

$$V_i(e(k)) \le e^T(k)Qe(k) + u_e^T(k)Ru_e(k) + V_{\infty}(e(k+1)). \tag{67}$$

Let $i \to \infty$. Then, we can obtain

$$V_{\infty}(e(k)) \le e^T(k)Qe(k) + u_e^T(k)Ru_e(k) + V_{\infty}(e(k+1)). \tag{68}$$

Note that $u_e(k)$ is chosen arbitrarily, so we have

$$V_{\infty}(e(k)) \le \min_{u_e(k)}\{e^T(k)Qe(k) + u_e^T(k)Ru_e(k) + V_{\infty}(e(k+1))\}. \tag{69}$$

On the other hand, for any $i$, the cost function sequence satisfies

$$V_i(e(k)) = \min_{u_e(k)}\{e^T(k)Qe(k) + u_e^T(k)Ru_e(k) + V_{i-1}(e(k+1))\}. \tag{70}$$

With $V_i(e(k)) \le V_{\infty}(e(k))$, $\forall i$, we have

$$V_{\infty}(e(k)) \ge \min_{u_e(k)}\{e^T(k)Qe(k) + u_e^T(k)Ru_e(k) + V_{i-1}(e(k+1))\}. \tag{71}$$

Let $i \to \infty$. Then we can get

$$V_{\infty}(e(k)) \ge \min_{u_e(k)}\{e^T(k)Qe(k) + u_e^T(k)Ru_e(k) + V_{\infty}(e(k+1))\}. \tag{72}$$

Then, combining (69) and (72), we can obtain (65). □

As a result, we can conclude that the cost function sequence $\{V_i\}$ converges to the optimal cost function of the discrete-time HJB equation $J^*$, i.e., $V_i \to J^*$ as $i \to \infty$. Furthermore, according to (41), it is clear that the corresponding control law sequence $\{v_i\}$ converges to the optimal control law $u_e^*$, i.e., $v_i \to u_e^*$ as $i \to \infty$.

### 4.3. NN implementation of the iterative ADP algorithm

In this part, we implement the iterative HDP algorithm using NNs. Since the regulation system (8) converted from the tracking control problem is still a nonlinear system, it is difficult to get the optimal feedback control analytically. Thus, we need to use parametric structures, such as NNs, to approximate the cost function and the corresponding control law.

In the iterative HDP algorithm, there are four NNs, which are model network (the NN identifier), the feedforward neuro-controller, critic network and action network. All the networks are chosen as three-layer feedforward NNs. The structure diagram of the iterative HDP algorithm is shown in Fig. 2, where $\hat{e}(k+1) = \hat{x}(k+1) - r(k+1)$.

From Fig. 2, it can be seen that with $e(k)$, $r(k)$ and $\hat{v}_i(e(k))$, the estimated tracking error $\hat{e}(k+1)$ can be obtained with the aid of the NN identifier and feedforward neuto-controller. Using the feedforward neuro-controller, we can obtain the desired control $u_d(k)$ corresponding to the reference trajectory $r(k)$. Then, with $x(k) = e(k) + r(k)$ and $u(k) = \hat{v}_i(e(k)) + u_d(k)$, $\hat{x}(k+1)$ can be obtained by the NN identifier. Furthermore, with $\hat{x}(k+1)$ and $r(k+1)$, we can get the estimated tracking error $\hat{e}(k+1)$. The detailed establishments of the NN identifier and feedforward neuro-controller have been shown in Section 3.

Additionally, it should be noted that before implementing the iterative algorithm, the training of both the NN identifier and the feedforward neuro-controller should be completed. Then, the corresponding weights are kept unchanged.

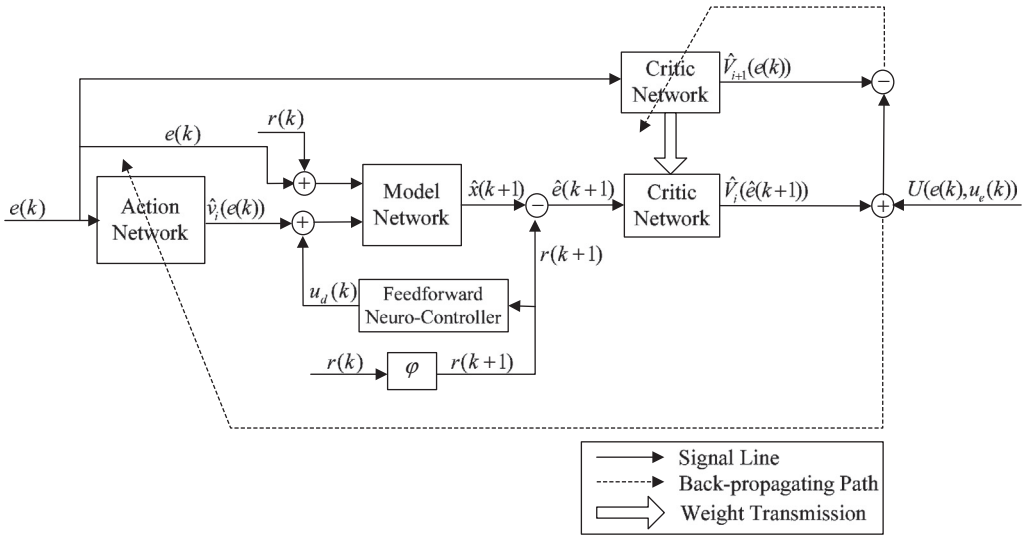


**Fig. 2.** The structure diagram of the iterative HDP algorithm.

The critic network is used for approximating the cost function $V_{i+1}(e(k))$. The output of the critic network is denoted as

$$\hat{V}_{i+1}(e(k)) = \omega_{c(i+1)}^T \sigma(v_{c(i+1)}^T e(k)). \tag{73}$$

Based on (42), $V_{i+1}(e(k))$ can be written as

$$V_{i+1}(e(k)) = e^T(k)Qe(k) + \hat{v}_i^T(e(k))R\hat{v}_i(e(k)) + \hat{V}_i(e(k+1)). \tag{74}$$

Thus, we define the prediction error for the critic network as

$$e_{c(i+1)}(k) = \hat{V}_{i+1}(e(k)) - V_{i+1}(e(k)). \tag{75}$$

The objective function to be minimized for the critic network is

$$E_{c(i+1)}(k) = \tfrac{1}{2} e_{c(i+1)}^T(k) e_{c(i+1)}(k). \tag{76}$$

The weight updating rule for the critic network is the gradient-based adaptation given by

$$\omega_{c(i+1)}(j+1) = \omega_{c(i+1)}(j) - \alpha_c \left[\frac{\partial E_{c(i+1)}(k)}{\partial \omega_{c(i+1)}(j)}\right],$$

$$v_{c(i+1)}(j+1) = v_{c(i+1)}(j) - \alpha_c \left[\frac{\partial E_{c(i+1)}(k)}{\partial v_{c(i+1)}(j)}\right], \tag{77}$$

where $\alpha_c > 0$ is the learning rate of the critic network and $j$ is the inner-loop iterative step for updating the weight parameters.

The action network takes the state $e(k)$ as the input and outputs the control $\hat{v}_i(e(k))$. The output can be formulated as

$$\hat{v}_i(e(k)) = \omega_{ci}^T \sigma(v_{ci}^T e(k)). \tag{78}$$

Considering (41), the target value of the control $v_i(k)$ can be obtained by

$$v_i(e(k)) = -\frac{1}{2} R^{-1} g^T(e(k) + r(k)) \frac{\partial \hat{V}_i(e(k+1))}{\partial \hat{e}(k+1)}. \tag{79}$$

Thus, we define the prediction error for the action network as

$$e_{ai}(k) = \hat{v}_i(e(k)) - v_i(e(k)). \tag{80}$$

The weights of the action network are updated to minimize the following performance error measure:

$$E_{ai}(k) = \tfrac{1}{2} e_{ai}^T(k) e_{ai}(k). \tag{81}$$

The weight update for the action network is similar to the weight adjustment in the critic network. By the gradient descent rule

$$\omega_{ai}(j+1) = \omega_{ai}(j) - \beta_a \left[\frac{\partial E_{ai}(k)}{\partial \omega_{ai}(j)}\right],$$

$$v_{ai}(j+1) = v_{ai}(j) - \beta_a \left[\frac{\partial E_{ai}(k)}{\partial v_{ai}(j)}\right], \tag{82}$$

where $\beta_a > 0$ is the learning rate of the action network and $j$ is the inner-loop iterative step for updating the weight parameters.

Additionally, considering (79), it is important to note that the control coefficient matrix $g(x(k))$ is required for computing the target value of the control $v_i(e(k))$. However, for unknown nonlinear systems, $g(x(k))$ cannot be obtained directly. In this paper, using the NN identifier, we can obtain the estimated value of the control coefficient matrix, i.e., $\hat{g}(x(k))$. Considering (1) and (33), we have

$$\begin{aligned}\hat{g}(x(k)) &= \frac{\partial \hat{\omega}_m^T(k)\sigma(\bar{z}(k))}{\partial u(k)} \\ &= \hat{\omega}_m^T(k) \frac{\partial \sigma(\bar{z}(k))}{\partial \bar{z}(k)} v_m^{*T} \frac{\partial z(k)}{\partial u(k)}.\end{aligned} \tag{83}$$

Thus, it is clear that $\hat{g}(x(k))$ can be obtained by the backpropagation from the outputs $\hat{x}(k+1)$ of the NN identifier to its input $u(k)$.

## 5. Simulation study

In this section, an example is provided to demonstrate the effectiveness of the proposed tracking control scheme. The example is derived from [24].

Considering the following nonlinear system:

$$x(k+1) = f(x(k)) + g(x(k))u(k), \tag{84}$$

where $x(k) = [x_1(k)\ x_2(k)]^T \in \mathbb{R}^2$ is the system state and $u(k) = [u_1(k)\ u_2(k)]^T \in \mathbb{R}^2$ is the control input. The corresponding $f(x(k))$ and $g(x(k))$ are given as

$$f(x(k)) = \begin{bmatrix} -\sin(0.5x_2(k))x_1^2(k) \\ -\cos(1.4x_2(k))\sin(0.9x_1(k)) \end{bmatrix},$$

$$g(x(k)) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

The reference trajectory for the above system is defined as

$$r(k) = \begin{bmatrix} \sin(0.25k) \\ \cos(0.25k) \end{bmatrix}.$$

The initial state is set as $x(0)=[0.5\ 0.5]^T$. The cost function is defined by (10), where $Q$ and $R$ are chosen as identity matrices of appropriate dimensions. It is assumed that the system dynamics is completely unknown and the input–output data are available.

To begin with, an NN identifier is established by a three-layer BP NN which is chosen as 4–8–2 structure with 4 input neurons, 8 hidden neurons and 2 output neurons. For the NN identifier, the hidden layer uses the sigmoidal function *tansig* and the output layer uses the linear function *purelin*. Let the learning rate $l_m=0.05$. With the initial weights being chosen randomly in $[-0.1,0.1]$, we train the NN identifier for 1000 steps using 500 data samples by the Levenberg–Marquardt algorithm. The training result is shown in Fig. 3.

From Fig. 3, it is clear that since the system identification error converges to zero, so the NN identifier successfully learns the dynamics of the unknown nonlinear system. Then, with the aid of the trained NN identifier, a feedforward neuro-controller is set up by a BP NN with the structure of 2–15–2. Similar to the NN identifier, the initial weights are chosen randomly in $[-0.1,0.1]$. Considering the practical tracking region, the reference output is given randomly in the time range from 0 to 1000 seconds in our simulation. Then, we train the feedforward neuro-controller for 1000 steps using 1000 data samples. After the training of the feedforward neuro-controller completes, it is tested with a reference output within the selected time region. The results are shown in Figs. 4 and 5.

From Figs. 4 and 5, it can be seen that the feedforward neuro-controller have learned the desired control $u_d(k)$.

The initial weights of the critic and action networks are all set to be random in $[-0.1,0.1]$. Let the learning rate $\alpha_c=\beta_a=0.05$, we train the critic and action networks for 200 iterations (i.e., for $i=1,2,\ldots,200$) with each of iteration containing 300 training steps to reach the given accuracy $\varepsilon=10^{-6}$. Then, we apply the optimal tracking control scheme to the system for 50 time steps and obtain the relevant simulation results. The obtained state curves are shown in Figs. 6 and 7, where the corresponding reference trajectories are also plotted for assessing the tracking performance. The tracking control curves and the tracking errors are shown in Figs. 8 and 9, respectively. The simulation results show that the proposed tracking controller based on the iterative HDP algorithm obtains the satisfying tracking performance for unknown nonlinear systems.

To further check the effectiveness of the proposed optimal tracking control scheme, for the above example, we change the control coefficient matrix $g(x(k))$ from a constant matrix to a function matrix.

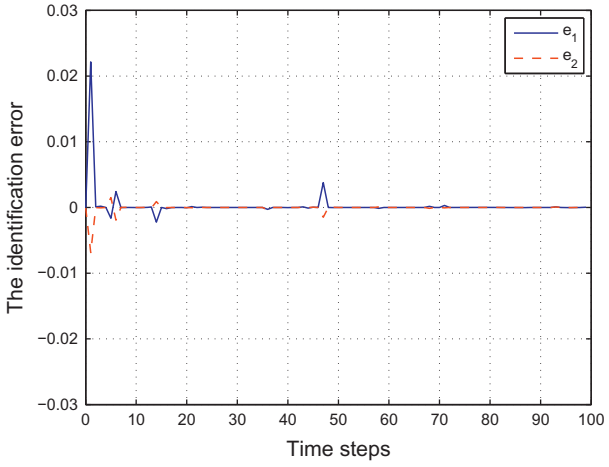


**Fig. 3.** The system identification error.

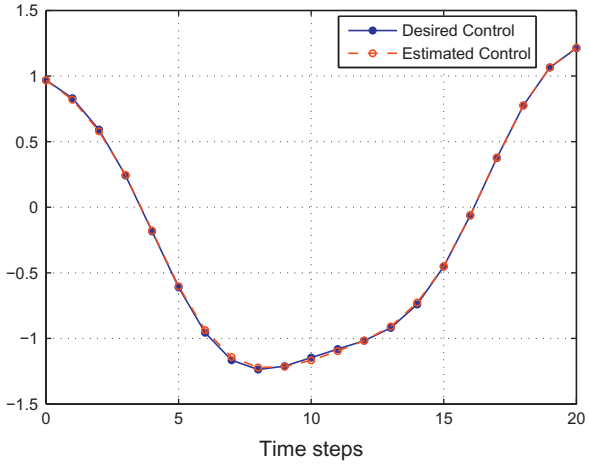


**Fig. 4.** The desired control $u_{d1}$ from feedforward neuro-controller.

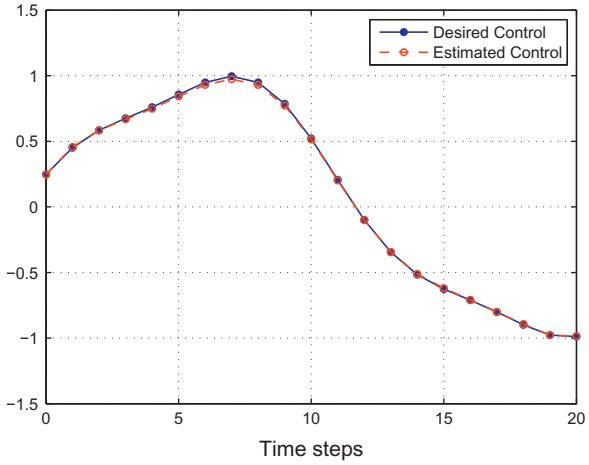


**Fig. 5.** The desired control $u_{d2}$ from feedforward neuro-controller.

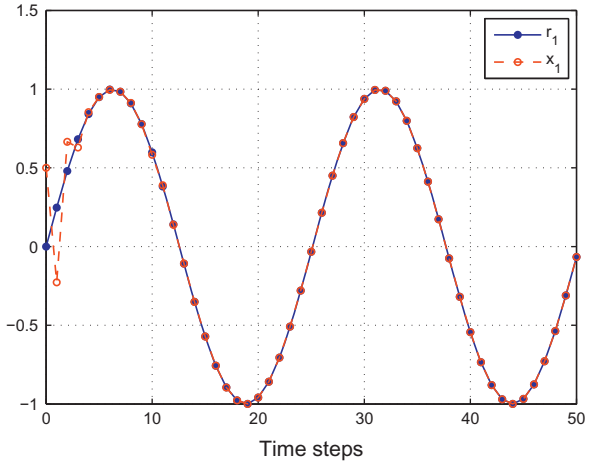


**Fig. 6.** The state trajectory $x_1$ and the reference trajectory $r_1$.

Let the function matrix $g(x(k))$ be rewritten as

$$g(x(k))=\begin{bmatrix}(x_1(k))^2+1.5 & 0.1\\ 0 & 0.2((x_1(k)+x_2(k))^2+1)\end{bmatrix}.$$

All the other parameters are set the same as the above.

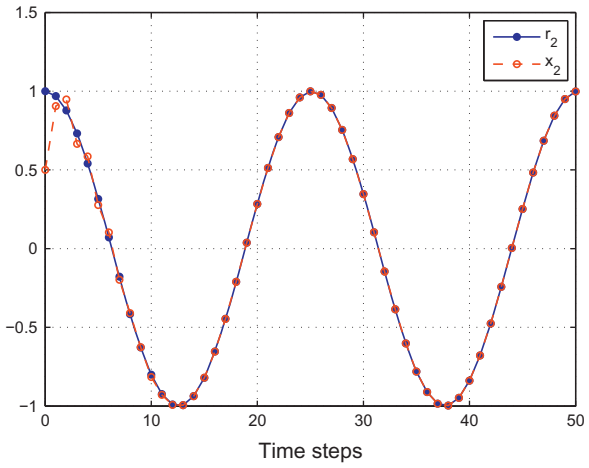


**Fig. 7.** The state trajectory $x_2$ and the reference trajectory $r_2$.

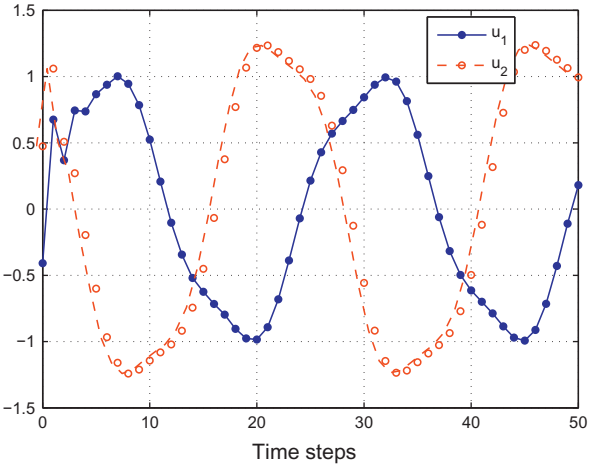


**Fig. 8.** The tracking control trajectory $u$.

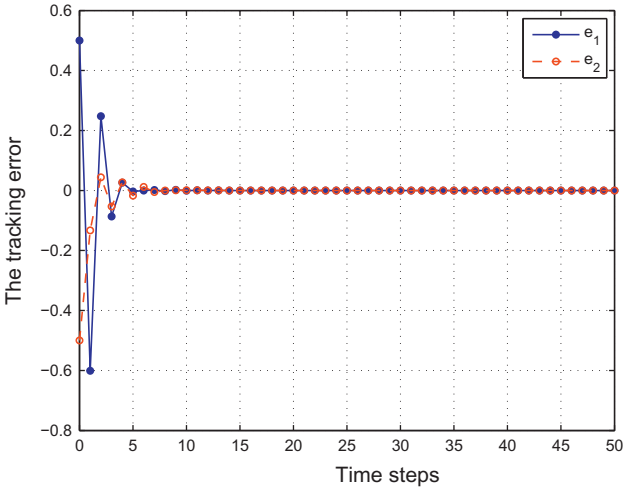


**Fig. 9.** The tracking error $e$.

First, the NN identifier and feedforward neuro-controller are retrained using the new input–output data with the relevant NN parameters selected the same as in the above example. Then, let the initial weights of the critic and action networks be random in $[-0.1,0.1]$. With the learning rate $\alpha_c = \beta_a = 0.05$, we train the critic and action networks for 300 iterations (i.e., for $i = 1,2,\ldots,300$) with each of iteration containing 500 training steps to reach the given accuracy $\varepsilon = 10^{-6}$. Then we apply the optimal tracking control scheme to the system for 50 time steps and obtain the relevant simulation results.

The obtained state curves are shown in Figs. 10 and 11. The tracking errors are shown in Fig. 12. Besides, the tracking control curves are given in Fig. 13. Hence, from these simulation results, it is clear that the optimal tracking control scheme proposed in this paper is very effective in solving the tracking control problems for unknown nonlinear systems.

## 6. Conclusion

In this paper, we propose an optimal tracking control scheme based on the iterative HDP algorithm for a class of unknown discrete-time nonlinear systems. For dealing with the unknown nonlinear system, two BP NNs are employed to construct the NN identifier and feedforward neuro-controller, respectively. The NN identifier is used to learn the dynamics of the system, while the feedforward neuro-controller is used to learn the inverse dynamics of the system for outputting the desired tracking control. Then, the iterative HDP algorithm is introduced to obtain the optimal feedback control for stabilizing the state tracking error dynamics. In the implementation of the iterative algorithm,

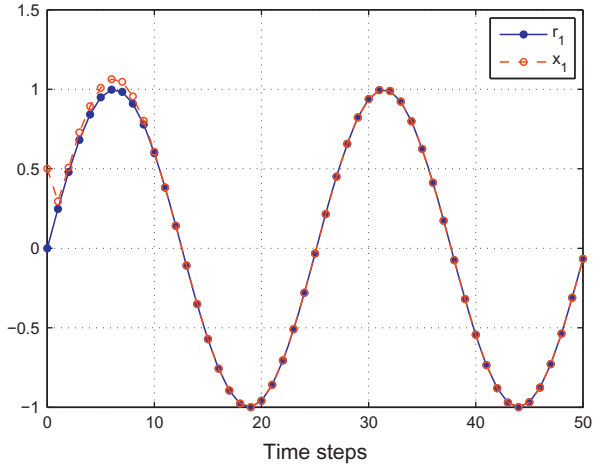


**Fig. 10.** The state trajectory $x_1$ and the reference trajectory $r_1$.

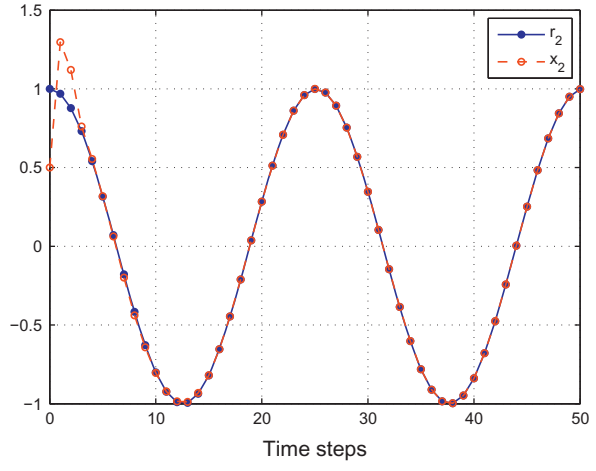


**Fig. 11.** The state trajectory $x_2$ and the reference trajectory $r_2$.

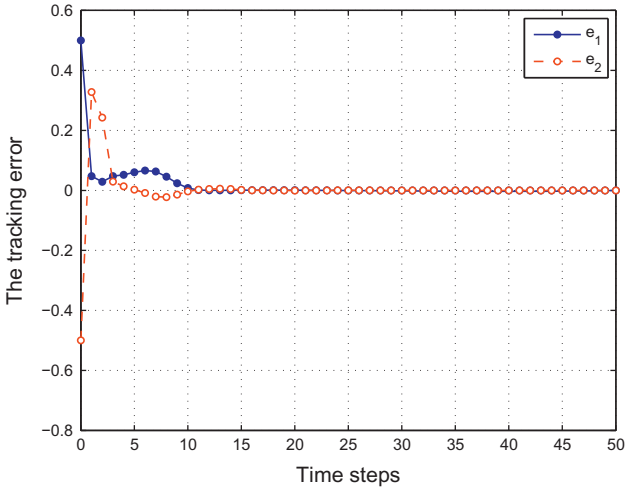


**Fig. 12.** The tracking error $e$.

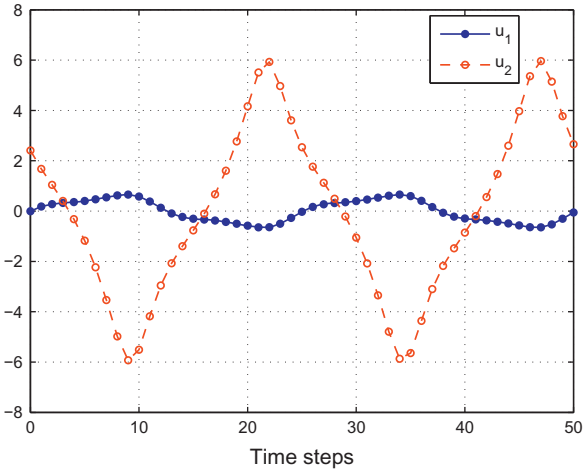


**Fig. 13.** The tracking control trajectory $u$.

the feedforward NNs are used as parametric structures to approximate the cost function and the corresponding control. Simulation results confirmed the validity of the optimal tracking controller based on the iterative HDP algorithm.

## Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grants 60904037, 60921061, and 61034002, in part by Beijing Natural Science Foundation under Grant 4102061, and in part by China Postdoctoral Science Foundation under Grant 201104162.

## References

[1] I.J. Ha, E.G. Gilbert, Robust tracking in nonlinear systems, IEEE Trans. Autom. Control 32 (1987) 763–771.
[2] S. Devasiad, D. Chen, B. Paden, Nonlinear inversion-based output tracking, IEEE Trans. Autom. Control 41 (1996) 930–942.
[3] T.W. Mclain, C.A. Bailry, R.W. Beard, Synthesis and experimental testing of a nonlinear optimal tracking controller, in: Proceedings of American Control Conference, San Diego, CA, June 1999, pp. 2847–2851.
[4] L. Cui, H. Zhang, B. Chen, Q. Zhang, Asymptotic tracking control scheme for mechanical systems with external disturbances and friction, Neurocomputing 73 (2010) 1293–1302.
[5] R. Song, H. Zhang, Y. Luo, Q. Wei, Optimal control laws for time-delay systems with saturating actuators based on heuristic dynamic programming, Neurocomputing 73 (2010) 3020–3027.
[6] V. Yadav, R. Padhi, S.M. Balakrishnan, Robust/optimal temperature profile control of a high-speed aerospace vehicle using neural networks, IEEE Trans. Neural Networks 18 (2007) 1115–1128.
[7] R.E. Bellman, Dynamic Programming, Princeton University Press, Princeton, NJ, 1957.
[8] F.L. Lewis, V.L. Syrmos, Optimal Control, Wiley, New York, 1995.
[9] S. Jagannathan, Neural Network Control of Nonlinear Discrete-Time Systems, CRC Press, Boca Raton, FL, 2006.
[10] P.J. Werbos, Advanced forecasting methods for global crisis warning and models of intelligence, Gen. Syst. Yearb. 22 (1977) 25–38.
[11] P.J. Werbos, Approximate dynamic programming for real-time control and neural modeling, in: D.A. White, D.A. Sofge (Eds.), Handbook of Intelligent Control, Van Nostrand Reinhold, New York, 1992. (Chapter 13).
[12] P.J. Werbos, Using ADP to understand and replicate brain intelligence: the next level design, in: Proceedings of the IEEE Symposium on Approximate Dynamic Programming and Reinforcement Learning, Honolulu, HI, April 2007, pp. 209–216.
[13] P.J. Werbos, ADP: the key direction for future research in intelligent control and understanding brain intelligence, IEEE Trans. Syst. Man Cybern. Part B Cybern. 38 (2008) 898–900.
[14] F.Y. Wang, H. Zhang, D. Liu, Adaptive dynamic programming: an introduction, IEEE Comput. Intell. Mag. 4 (2009) 39–47.
[15] F.L. Lewis, D. Vrabie, Reinforcement learning and adaptive dynamic programming for feedback control, IEEE Circuits Syst. Mag. 9 (2009) 32–50.
[16] D.P. Bertsekas, J.N. Tsitsiklis, Neuro-Dynamic Programming, Athena Scientific, Belmont, MA, 1996.
[17] D.V. Prokhorov, R.A. Santiago, D.C. Wunsch, Adaptive critic designs: a case study for neuro-control, Neural Networks 8 (1995) 1367–1372.
[18] D.V. Prokhorov, D.C. Wunsch, Adaptive critic designs, IEEE Trans. Neural Networks 8 (1997) 997–1007.
[19] A. Al-Tamimi, F.L. Lewis, M. Abu-Khalaf, Discrete-time nonlinear HJB solution using approximate dynamic programming: convergence proof, IEEE Trans. Syst. Man Cybern. Part B Cybern. 38 (2008) 943–949.
[20] D. Vrabie, F.L. Lewis, Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems, Neural Networks 22 (2009) 237–246.
[21] F.Y. Wang, N. Jin, D. Liu, Q. Wei, Adaptive dynamic programming for finite-horizon optimal control of discrete-time nonlinear systems with $\varepsilon$-error bound, IEEE Trans. Neural Networks 22 (2011) 24–36.
[22] T. Dierks, B.T. Thumati, S. Jagannathan, Optimal control of unknown affine nonlinear discrete-time systems using offline-trained neural networks with proof of convergence, Neural Networks 22 (2009) 851–860.
[23] Y.M. Park, M.S. Choi, K.Y. Lee, An optimal tracking neuro-controller for nonlinear dynamic systems, IEEE Trans. Neural Networks 7 (1996) 1099–1110.
[24] T. Dierks, S. Jagannathan, Optimal tracking control of affine nonlinear discrete-time systems with unknown internal dynamics, in: Proceedings of Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference, Shanghai, PR China, December 2009, pp. 6750–6755.
[25] H. Zhang, Q. Wei, Y. Luo, A novel infinite-time optimal tracking control scheme for a class of discrete-time nonlinear system via the greedy HDP iteration algorithm, IEEE Trans. Syst. Man Cybern. Part B Cybern. 38 (2008) 937–942.
[26] D. Wang, D. Liu, Q. Wei, Finite-horizon neuro-optimal tracking control for a class of discrete-time nonlinear systems using adaptive dynamic programming approach, Neurocomputing 78 (2012) 14–22.
[27] J.J. Murray, C.J. Cox, G.G. Lendaris, R. Saeks, Adaptive dynamic programming, IEEE Trans. Syst. Man Cybern. Part C Appl. Rev. 32 (2002) 140–153.
[28] J. Si, A.G. Barto, W.B. Powell, D.C. Wunsch (Eds.), Handbook of Learning and Approximate Dynamic Programming, IEEE Press, Wiley, New York, 2004.
[29] J. Si, Y.T. Wang, On-line learning control by association and reinforcement, IEEE Trans. Neural Networks 12 (2001) 264–276.
[30] H. Zhang, Y. Luo, D. Liu, Neural-network-based near-optimal control for a class of discrete-time affine nonlinear systems with control constraints, IEEE Trans. Neural Networks 9 (2009) 1490–1503.
[31] D. Liu, N. Jin, $\varepsilon$-Adaptive dynamic programming for discrete-time systems, in: Proceedings of International Joint Conference on Neural Networks, Hong Kong, June 2008, pp. 1417–1424.
[32] D. Liu, Y. Zhang, H. Zhang, A self-learning call admission control scheme for CDMA cellular networks, IEEE Trans. Neural Networks 16 (2005) 1219–1228.
[33] D. Liu, X. Xiong, Y. Zhang, Action-dependent adaptive critic designs, in: Proceedings of International Joint Conference on Neural Networks, Washington, DC, July 2001, pp. 990–995.
[34] X. Zhang, H. Zhang, Q. Sun, Y. Luo, Adaptive dynamic programming-based optimal control of unknown nonaffine discrete-time systems with proof of convergence, Neurocomputing 91 (2012) 48–55.
[35] M. Abu-Khalaf, F.L. Lewis, Nearly optimal control laws for nonlinear systems with saturating actuators using a neural network HJB approach, Automatica 41 (2005) 779–791.
[36] S.N. Balakrishnan, J. Ding, F.L. Lewis, Issues on stability of ADP feedback controllers for dynamic systems, IEEE Trans. Syst. Man Cybern. Part B Cybern. 38 (2008) 913–917.

[37] P. He, S. Jagannathan, Reinforcement learning neural-network-based controller for nonlinear discrete-time systems with input constraints, IEEE Trans. Syst. Man Cybern. Part B Cybern. 37 (2007) 425–436.
[38] G.K. Venayagamoorthy, R.G. Harley, D.C. Wunsch, Implementation of adaptive critic-based neurocontrollers for turbogenerators in a multimachine power system, IEEE Trans. Neural Networks 14 (2003) 1047–1064.
[39] G.K. Venayagamoorthy, R.G. Harley, D.C. Wunsch, Comparison of heuristic dynamic programming and dual heuristic programming adaptive critics for neurocontrol of a turbogenerator, IEEE Trans. Neural Networks 13 (2002) 764–773.

**Yuzhu Huang** received the B.S. degree in automation from Auhui University of Science and Technology, Huainan, China, and the M.S. degree in control theory and control engineering from Beijing Institute of Technology, Beijing, China, in 2008 and 2010, respectively. He is currently working toward the Ph.D. degree in The State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, China. His research interests include adaptive dynamic programming, neural networks, game theory and intelligent control.

**Derong Liu** received the Ph.D. degree in electrical engineering from the University of Notre Dame in 1994. He was a Staff Fellow with General Motors Research and Development Center, Warren, MI, from 1993 to 1995. He was an Assistant Professor in the Department of Electrical and Computer Engineering, Stevens Institute of Technology, Hoboken, NJ, from 1995 to 1999. He joined the University of Illinois at Chicago in 1999, and became a Full Professor of electrical and computer engineering and of computer science in 2006. He was selected for the "100 Talents Program" by the Chinese Academy of Sciences in 2008. He has published 13 books. Dr. Liu has been an Associate Editor of several IEEE publications. Currently, he is the Editor-in-Chief of the IEEE Transactions on Neural Networks and Learning Systems, and an Associate Editor of the IEEE Transactions on Control Systems Technology and several other journals including Neurocomputing, International Journal of Neural Systems, Neural Computing and Applications, Soft Computing, Journal of Control Science and Engineering, and Science in China Series F: Information Sciences. He was an elected AdCom member of the IEEE Computational Intelligence Society (2006–2008). He received the Faculty Early Career Development (CAREER) award from the National Science Foundation (1999), the University Scholar Award from University of Illinois (2006–2009), and the Overseas Outstanding Young Scholar Award from the National Natural Science Foundation of China (2008). He is a Fellow of the IEEE.