Finite-Approximation-Error-Based Optimal Control Approach for Discrete-Time Nonlinear Systems

Derong Liu, *Fellow, IEEE*, and Qinglai Wei

***Abstract*—In this paper, a new iterative adaptive dynamic programming (ADP) algorithm is developed to solve optimal control problems for infinite-horizon discrete-time nonlinear systems with finite approximation errors. The idea is to use an iterative ADP algorithm to obtain the iterative control law that makes the iterative performance index function reach the optimum. When the iterative control law and the iterative performance index function in each iteration cannot be accurately obtained, the convergence conditions of the iterative ADP algorithm are obtained. When convergence conditions are satisfied, it is shown that the iterative performance index functions can converge to a finite neighborhood of the greatest lower bound of all performance index functions under some mild assumptions. Neural networks are used to approximate the performance index function and compute the optimal control policy, respectively, for facilitating the implementation of the iterative ADP algorithm. Finally, two simulation examples are given to illustrate the performance of the present method.**

***Index Terms*—Adaptive dynamic programming (ADP), approximate dynamic programming, finite approximation errors, neural networks, optimal control.**

I. Introduction

OPTIMAL control of nonlinear systems has always been the key focus in the control field in the latest several decades [12], [19], [21]. Dynamic programming is a very useful tool in solving optimal control problems. However, due to the difficulties in solving nonlinear two-point boundary value problems or the time-varying Hamilton–Jacobi–Bellman (HJB) equations, analytical forms of the optimal control solutions for nonlinear systems are usually impossible to obtain. Approximate solutions of optimal control problems have attracted a lot of attention [3]–[5], [8], [20]. Among these approximate approaches, adaptive dynamic programming (ADP) algorithm, proposed by Werbos [33], has played an important role in seeking approximate solutions of dynamic programming problems as a way to solve the computational issue forward-in-time [18], [23], [24]. There are several synonyms used for ADP, including "adaptive critic designs" [22], "adaptive dynamic programming" [20], [28], "approximate dynamic programming" [2], [13], [34], "neural dynamic programming" [7], "neurodynamic programming" [5], and "reinforcement learning" [14]. In [22] and [34], ADP approaches were classified into several main schemes: heuristic dynamic programming (HDP), dual heuristic programming (DHP), action-dependent HDP (also known as Q-learning [29]), action-dependent DHP, globalized DHP (GDHP), and action-dependent GDHP. Recursive methods are also used in ADP to obtain the solution of HJB equation indirectly and have received lots of attention. There are two main recursive ADP algorithms that are based on policy and value iterations, respectively [15].

Policy iteration algorithms for optimal control of continuous-time systems with continuous state and action spaces were given in [1]. In 2011, Wang *et al.* [27] studied finite-horizon optimal control problems for discrete-time nonlinear systems with unspecified terminal time. In the policy iteration algorithms of ADP, to obtain the performance index functions and control actions iteratively, an initial admissible control sequence of the system must be required. However, unfortunately, the admissible control sequence for the nonlinear system is also difficult to obtain. Thus, the initial conditions for the controller greatly limit the applications of the policy iteration algorithms. On the other hand, value iteration algorithms for optimal control of discrete-time nonlinear systems were given in [4]. In [2], a value iteration algorithm, which is referred to as greedy HDP iteration algorithm, was proposed. The convergence of the algorithm was proved in [2] and [16]. In [36], an optimal tracking control problem for a class of nonlinear discrete-time systems was solved by a value iteration algorithm. For the value iteration algorithms of ADP, the initial admissible control sequence of the nonlinear system can be avoided, whereas the stability property of control systems under the iterative control cannot be guaranteed. This means that the value iteration algorithm can be only implemented offline. In 2012, Wei and Liu [30], [31] proposed a new iterative θ-ADP algorithm for discrete-time nonlinear systems. In the iterative θ-ADP algorithm, the initial admissible control sequence is unnecessary, and it is proved that each of the iterative controls is stable for the nonlinear system. This makes the iterative θ-ADP algorithm feasible for implementation both online and offline.

Although iterative ADP algorithms attract more and more attention [11], [17], [26], [32], [35], [37], for nearly all of the iterative algorithms, the iterative control of each iteration is required to be accurately obtained. These iterative ADP algorithms can be called "accurate iterative ADP algorithms." For most real-world control systems, however, accurate iterative control laws in the iterative ADP algorithms cannot be obtained. For example, during the implementation

Manuscript received January 8, 2012; revised July 27, 2012; accepted August 14, 2012. Date of publication October 10, 2012; date of current version April 16, 2013. This work was supported in part by the National Natural Science Foundation of China under Grant 60904037, Grant 60921061, and Grant 61034002; by Beijing Natural Science Foundation under Grant 4102061; and by China Postdoctoral Science Foundation under Grant 201104162. This paper was recommended by Associate Editor H. Zhang.

The authors are with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China (e-mail: derong.liu@ia.ac.cn; qinglai.wei@ia.ac.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCB.2012.2216523

of iterative ADP algorithms, approximation structures such as neural networks and fuzzy systems are used to approximate the iterative performance index functions and the iterative control laws. While we can see that no matter what kind of neural networks and fuzzy systems is used and no matter what approximation precision values are obtained, there always exist approximation errors between the approximation functions and the expected ones. This shows that the accurate performance index function and control laws cannot be reached in the iterative ADP algorithms for real-world control systems. When the accurate iterative control laws cannot be obtained, the convergence properties in the accurate iterative ADP algorithms may be invalid. Until now, to the best of our knowledge, there are no discussions on the convergence properties of the iterative ADP algorithms when the accurate iterative control cannot be obtained, which limits the applications of ADP in general. This motivates our research.

In this paper, we will develop a new ADP scheme for infinite-horizon optimal control problems. The main contribution of this paper is that the optimal control problems with finite approximation errors are effectively solved using the proposed iterative ADP algorithms. First, the HJB equation for infinite-horizon optimal control of discrete-time systems is derived. Second, for the situation that accurate iterative performance index functions and iterative control laws cannot be obtained, in order to solve the HJB equation, a new iterative ADP algorithm with finite approximation errors is developed with convergence and optimality proofs. Next, it will be shown that the proposed iterative ADP algorithm can make the iterative performance index functions converge to a finite neighborhood of the optimal performance index function if the convergence condition is satisfied. A new convergence analysis method is proposed, and the least upper bound of the converged iterative performance index function is also obtained. Furthermore, in order to facilitate the implementation of iterative ADP algorithms, we use neural networks to obtain the iterative performance index function and the optimal control policy. Finally, two simulation examples are given to show the effectiveness of the proposed iterative ADP algorithm.

This paper is organized as follows. In Section II, the problem statement is presented. In Section III, the iterative ADP algorithm with finite approximation errors is derived. In Section IV, the neural network implementation for the optimal control scheme is discussed. In Section V, two simulation examples are given to demonstrate the effectiveness of the proposed control scheme. Finally, in Section VI, this paper is concluded with a few pertinent remarks.

II. Problem Statement

In this paper, we will study the following deterministic discrete-time system:

$$x_{k+1} = F(x_k, u_k), \quad k = 0, 1, 2, \ldots \tag{1}$$

where $x_k \in \mathbb{R}^n$ is the n-dimensional state vector, and $u_k \in \mathbb{R}^m$ is the m-dimensional control vector. Let x_0 be the initial state and $F(x_k, u_k)$ be the system function.

Let $\underline{u}_k = (u_k, u_{k+1}, \ldots)$ be an arbitrary sequence of controls from k to ∞. The performance index function for state x_0 under the control sequence $\underline{u}_0 = (u_0, u_1, \ldots)$ is defined as

$$J(x_0, \underline{u}_0) = \sum_{k=0}^{\infty} U(x_k, u_k) \tag{2}$$

where $U(x_k, u_k) > 0, \forall\, x_k, u_k$, is the utility function.

In this paper, we will study optimal control problems for (1). The goal of this paper is to find an optimal control scheme that stabilizes system (1) and simultaneously minimizes the performance index function (2). For convenience of analysis, results in this paper are based on the following assumptions.

Assumption 1: System (1) is controllable, and the function $F(x_k, u_k)$ is Lipschitz continuous for $\forall\, x_k, u_k$.

Assumption 2: The system state $x_k = 0$ is an equilibrium state of system (1) under the control $u_k = 0$, i.e., $F(0, 0) = 0$.

Assumption 3: The feedback control $u_k = u(x_k)$ satisfies $u_k = u(x_k) = 0$ for $x_k = 0$.

Assumption 4: The utility function $U(x_k, u_k)$ is a continuous positive definite function of x_k and u_k.

Define the control sequence set as $\underline{\mathfrak{U}}_k = \{\underline{u}_k : \underline{u}_k = (u_k, u_{k+1}, \ldots), \forall u_{k+i} \in \mathbb{R}^m, i = 0, 1, \ldots\}$. Then, for arbitrary control sequence $\underline{u}_k \in \underline{\mathfrak{U}}_k$, the optimal performance index function can be defined as

$$J^*(x_k) = \inf_{\underline{u}_k} \left\{ J(x_k, \underline{u}_k) : \underline{u}_k \in \underline{\mathfrak{U}}_k \right\}. \tag{3}$$

According to Bellman's principle of optimality, $J^*(x_k)$ satisfies the discrete-time HJB equation

$$J^*(x_k) = \min_{u_k \in \mathbb{R}^m} \left\{ U(x_k, u_k) + J^*\left(F(x_k, u_k)\right) \right\}. \tag{4}$$

Define the law of optimal control sequence as

$$\underline{u}^*(x_k) = \arg \inf_{\underline{u}_k} \left\{ J(x_k, \underline{u}_k) : \underline{u}_k \in \underline{\mathfrak{U}}_k \right\}. \tag{5}$$

Then, the law of optimal single control vector can be expressed as

$$u^*(x_k) = \arg \min_{u_k \in \mathbb{R}^m} \left\{ U(x_k, u_k) + J^*\left(F(x_k, u_k)\right) \right\}. \tag{6}$$

Hence, the HJB equation (4) can be written as

$$J^*(x_k) = U\left(x_k, u^*(x_k)\right) + J^*\left(F\left(x_k, u^*(x_k)\right)\right). \tag{7}$$

We can see that if we want to obtain the optimal control law $u^*(x_k)$, we must obtain the optimal performance index function $J^*(x_k)$. Generally, $J^*(x_k)$ is unknown before all the controls $u_k \in \mathbb{R}^m$ are considered. If we adopt the traditional dynamic programming method to obtain the optimal performance index function one step at a time, then we have to face "the curse of dimensionality." This makes the optimal control nearly impossible to be obtained by the HJB equation (7).

In [30] and [31], an iterative θ-ADP algorithm was proposed to obtain the optimal performance index function and the optimal control law iteratively. In the iterative θ-ADP algorithm, the performance index function and control law are updated by recurrent iteration, with iteration index i increasing from 0 to infinity.

For $\forall x_k \in \mathbb{R}^n$, let the initial function $\Psi(x_k)$ be an arbitrary function that satisfies $\Psi(x_k) \in \bar{\Psi}_{x_k}$, where $\bar{\Psi}_{x_k}$ is defined as follows.

Definition 2.1: Let

$$\bar{\Psi}_{x_k} = \{\Psi(x_k) : \Psi(x_k) > 0, \quad \text{and } \exists \nu(x_k) \in \mathbb{R}^m \\ \Psi\left(F\left(x_k, \nu(x_k)\right)\right) < \Psi(x_k)\} \quad (8)$$

be the initial positive definition function set.

It can be easily proved that $\bar{\Psi}_{x_k}$ is not an empty set. From the definition, we can see that all the Lyapunov functions belong to the initial positive definition function set $\bar{\Psi}_{x_k}$. For $\forall x_k \in \mathbb{R}^n$, let the initial performance index function $V_0(x_k) = \theta\Psi(x_k)$, where $\theta > 0$ is a large finite positive constant. The iterative control law $v_0(x_k)$ can be computed as follows:

$$\begin{aligned} v_0(x_k) &= \arg\min_{u_k \in \mathbb{R}^m} \{U(x_k, u_k) + V_0(x_{k+1})\} \\ &= \arg\min_{u_k \in \mathbb{R}^m} \{U(x_k, u_k) + V_0\left(F(x_k, u_k)\right)\} \end{aligned} \quad (9)$$

where $V_0(x_{k+1}) = \theta\Psi(x_{k+1})$. The performance index function can be updated as

$$V_1(x_k) = U\left(x_k, v_0(x_k)\right) + V_0\left(F\left(x_k, v_0(x_k)\right)\right). \quad (10)$$

For $i = 1, 2, \ldots$, the iterative ADP algorithm will iterate between

$$\begin{aligned} v_i(x_k) &= \arg\min_{u_k \in \mathbb{R}^m} \{U(x_k, u_k) + V_i(x_{k+1})\} \\ &= \arg\min_{u_k \in \mathbb{R}^m} \{U(x_k, u_k) + V_i\left(F(x_k, u_k)\right)\} \quad (11) \\ V_{i+1}(x_k) &= \min_{u_k \in \mathbb{R}^m} \{U(x_k, u_k) + V_i(x_{k+1})\} \\ &= U\left(x_k, v_i(x_k)\right) + V_i\left(F\left(x_k, v_i(x_k)\right)\right). \quad (12) \end{aligned}$$

For the iterative θ-ADP algorithm, we can see that for $\forall i = 0, 1, \ldots$, the accurate iterative control law and the accurate iterative performance index function must be obtained in order to guarantee the convergence of the iterative performance index function [30], [31]. In real-world implementations, however, for $\forall i = 0, 1, \ldots$, the accurate iterative control law $v_i(x_k)$ and the iterative performance index function $V_i(x_k)$ are generally impossible to obtain. For example, if neural networks are used to implement the iterative θ-ADP algorithm, no matter what kinds of neural networks we choose, the approximation error between the outputs of neural networks and the expected outputs will exist. Therefore, in this situation, the convergence of the iterative performance index function and the iterative control law may be invalid and the iterative ADP algorithm may be even divergent. To overcome this difficulty, a new ADP algorithm and analysis method will be developed.

III. Properties of the Iterative ADP Algorithm With Finite Approximation Errors

In the previous section, we have indicated that in the iterative θ-ADP algorithm (9)–(12), the iterative performance index function $V_i(x_k)$ converges to the optimal performance index function $J^*(x_k)$ and $J^*(x_k) = \inf_{\underline{u}_k}\{J(x_k, \underline{u}_k), \underline{u}_k \in \underline{\mathfrak{U}}_k\}$ satisfies the HJB equation (7) for any controllable state $x_k \in \mathbb{R}^n$. In fact, as the existence of the approximation errors, the accurate iterative control law cannot generally be obtained. In this situation, the iterative ADP algorithm with no approximation errors may be invalid. New analysis methods should be investigated.

Then, in the following, we give the iterative ADP algorithm with finite approximation errors.

A. Derivation of the Iterative ADP Algorithm With Finite Approximation Errors

In the proposed iterative ADP algorithm, the performance index function and control law are updated by recurrent iteration, with iteration index i increasing from 0 to infinity. For $\forall x_k \in \mathbb{R}^n$, let the initial function $\Psi(x_k)$ be an arbitrary function that satisfies $\Psi(x_k) \in \bar{\Psi}_{x_k}$, where $\bar{\Psi}_{x_k}$ is expressed in Definition 2.1. For $\forall x_k \in \mathbb{R}^n$, let the initial performance index function $\hat{V}_0(x_k) = \theta\Psi(x_k)$, where $\theta > 0$ is a large finite positive constant. The iterative control law $\hat{v}_0(x_k)$ can be computed as follows:

$$\begin{aligned} \hat{v}_0(x_k) &= \arg\min_{u_k \in \mathbb{R}^m} \left\{U(x_k, u_k) + \hat{V}_0(x_{k+1})\right\} + \rho_0(x_k) \\ &= \arg\min_{u_k \in \mathbb{R}^m} \left\{U(x_k, u_k) + \hat{V}_0\left(F(x_k, u_k)\right)\right\} + \rho_0(x_k) \end{aligned} \quad (13)$$

where $\hat{V}_0(x_{k+1}) = \theta\Psi(x_{k+1})$, and the performance index function can be updated as

$$\hat{V}_1(x_k) = U\left(x_k, \hat{v}_0(x_k)\right) + \hat{V}_0\left(F\left(x_k, \hat{v}_0(x_k)\right)\right) + \pi_0(x_k) \quad (14)$$

where $\rho_0(x_k)$ and $\pi_0(x_k)$ are the approximation error functions of the iterative control and iterative performance index function, respectively.

For $i = 1, 2, \ldots$, the iterative ADP algorithm will iterate between

$$\begin{aligned} \hat{v}_i(x_k) &= \arg\min_{u_k \in \mathbb{R}^m} \left\{U(x_k, u_k) + \hat{V}_i(x_{k+1})\right\} + \rho_i(x_k) \\ &= \arg\min_{u_k \in \mathbb{R}^m} \left\{U(x_k, u_k) + \hat{V}_i\left(F(x_k, u_k)\right)\right\} + \rho_i(x_k) \end{aligned} \quad (15)$$

$$\hat{V}_{i+1}(x_k) = U\left(x_k, \hat{v}_i(x_k)\right) + \hat{V}_i\left(F\left(x_k, \hat{v}_i(x_k)\right)\right) + \pi_i(x_k) \quad (16)$$

where $\rho_i(x_k)$ and $\pi_i(x_k)$ are the finite approximation error functions of the iterative control and iterative performance index function, respectively.

Remark 3.1: Due to the existence of approximation errors, we can see that the iterative performance index function $\hat{V}_i(x_k)$ may be smaller than zero for some large $\pi_i(x_k)$. Since the iterative performance index function $V_i(x_k) > 0$ holds for $\forall x_k \neq 0$, then $\hat{V}_i(x_k) \leq 0$ is meaningless. Thus, in this paper, it is assumed that $\hat{V}_i(x_k) > 0, \forall x_k \neq 0$.

Remark 3.2: From the iterative ADP algorithm (13)–(16), we can see that for $i = 0, 1, \ldots$, the iterative performance index function $V_i(x_k)$ and the iterative control law $v_i(x_k)$ in (9)–(12) are replaced by $\hat{V}_i(x_k)$ and $\hat{v}_i(x_k)$, respectively. Due to the existence of approximation errors, generally speaking, we have for $\forall i \geq 0$, $\hat{v}_i(x_k) \neq v_i(x_k)$ and $\hat{V}_{i+1}(x_k) \neq V_{i+1}(x_k)$. This means that there exists an error between $\hat{V}_{i+1}(x_k)$ and

$V_{i+1}(x_k)$. It should be pointed out that the iterative approximation is not a constant. The fact is that, as the iteration index $i \to \infty$, the boundary of iterative approximation errors will also increase to infinity, although in the single iteration, the approximation error is finite. The following theorem will show this property.

Theorem 3.1: Let $x_k \in \mathbb{R}^n$ be an arbitrary controllable state and Assumptions 1–4 hold. For $i = 1, 2, \ldots$, define a new iterative performance index function as

$$\Gamma_i(x_k) = \min_{u_k \in \mathbb{R}^m} \left\{ U(x_k, u_k) + \hat{V}_{i-1}(x_{k+1}) \right\} \tag{17}$$

where $\hat{V}_i(x_k)$ is defined in (16), and u_k can be accurately obtained in $\mathbb{R}^m$. Let the initial iterative performance index function $\hat{V}_0(x_k) = \Gamma_0(x_k) = \theta\Psi(x_k)$. If for $i = 1, 2, \ldots$ there exists a finite constant $\bar{\epsilon} \geq 0$ that makes

$$\left| \hat{V}_i(x_k) - \Gamma_i(x_k) \right| \leq \bar{\epsilon} \tag{18}$$

hold uniformly, then we have

$$\left| \hat{V}_i(x_k) - V_i(x_k) \right| \leq i\bar{\epsilon}. \tag{19}$$

Proof: The theorem can be proved by mathematical induction. First, let $i = 1$. We have

$$\begin{aligned} \Gamma_1(x_k) &= \min_{u_k \in \mathbb{R}^m} \left\{ U(x_k, u_k) + \hat{V}_0(x_{k+1}) \right\} \\ &= \min_{u_k \in \mathbb{R}^m} \left\{ U(x_k, u_k) + V_0\left(F(x_k, u_k)\right) \right\} \\ &= V_1(x_k). \end{aligned} \tag{20}$$

According to (18), we have

$$-\bar{\epsilon} \leq \hat{V}_i(x_k) - \Gamma_i(x_k) \leq \bar{\epsilon} \tag{21}$$

that holds for $\forall i = 1, 2, \ldots$. Then, we can get

$$-\bar{\epsilon} \leq \hat{V}_1(x_k) - V_1(x_k) \leq \bar{\epsilon} \tag{22}$$

which proves that

$$\left| \hat{V}_1(x_k) - V_1(x_k) \right| \leq \bar{\epsilon}. \tag{23}$$

Assume that (19) holds for $i = l - 1$, $l = 1, 2, \ldots$. Then, we have

$$-(l-1)\bar{\epsilon} \leq \hat{V}_{l-1}(x_k) - V_{l-1}(x_k) \leq (l-1)\bar{\epsilon}. \tag{24}$$

For $i = l$, we can get

$$\begin{aligned} \Gamma_l(x_k) &= \min_{u_k \in \mathbb{R}^m} \left\{ U(x_k, u_k) + \hat{V}_{l-1}(x_{k+1}) \right\} \\ &\geq \min_{u_k \in \mathbb{R}^m} \left\{ U(x_k, u_k) + V_{l-1}(x_{k+1}) - (l-1)\bar{\epsilon} \right\} \\ &= V_l(x_k) - (l-1)\bar{\epsilon} \end{aligned} \tag{25}$$

$$\begin{aligned} \Gamma_l(x_k) &= \min_{u_k \in \mathbb{R}^m} \left\{ U(x_k, u_k) + \hat{V}_{l-1}(x_{k+1}) \right\} \\ &\leq \min_{u_k \in \mathbb{R}^m} \left\{ U(x_k, u_k) + V_{l-1}(x_{k+1}) + (l-1)\bar{\epsilon} \right\} \\ &= V_l(x_k) + (l-1)\bar{\epsilon}. \end{aligned} \tag{26}$$

Then, according to (21), we can get (19) easily. ■

From Theorem 3.1, we can see that if we let a finite constant $-\bar{\epsilon} \leq \epsilon \leq \bar{\epsilon}$ that satisfies

$$\hat{V}_i(x_k) - \Gamma_i(x_k) \leq \epsilon \tag{27}$$

uniformly, then we can immediately obtain

$$\hat{V}_i(x_k) - V_i(x_k) \leq i\epsilon \tag{28}$$

where ϵ can be defined as a uniform finite approximation error (finite approximation error for brief). For the iterative θ-ADP algorithm (9)–(12), it has been proved that the iterative performance index function converges to the optimum as $i \to \infty$. From (28), we can see that if we let $\mathcal{T}_i = i\epsilon$ be the least upper bound of the iterative approximation errors, then for $\forall \epsilon \neq 0$, we have $\mathcal{T}_i \to \infty$ as $i \to \infty$. This means that, although the approximation error for each single step is finite and may be small, as the iteration index increases $i \to \infty$, it is possible that the approximation errors between $\hat{V}_i(x_k)$ and $V_i(x_k)$ increase to infinity. Hence, the convergence analysis theorems in [30] and [31] are invalid to justify the properties of $\hat{V}_i(x_k)$ and $\hat{v}_i(x_k)$ in (13)–(16). To overcome these difficulties, a new convergence analysis must be established.

B. Properties of the Iterative ADP Algorithm With Finite Approximation Errors

From the iterative ADP algorithm (13)–(16), we can see that for $\forall i = 0, 1, \ldots$, there exists an approximation error between the iterative performance index functions $\hat{V}_i(x_k)$ and $V_i(x_k)$. Furthermore, the detailed value of each iterative error is unknown and nearly impossible to obtain. It makes the properties of the iterative performance index function $\hat{V}_i(x_k)$ and the iterative control law $\hat{v}_i(x_k)$ very difficult to analyze. Therefore, in this section, a new "error bound" analysis method is proposed. The idea of the "error bound" analysis method is that, for each iterative index $i = 0, 1, \ldots$, the least upper bound of the iterative performance index functions $\hat{V}_i(x_k)$ is analyzed, which avoids analyzing the value of $\hat{V}_i(x_k)$ directly. Using the "error bound" method, it can be proved that the iterative performance index functions $\hat{V}_i(x_k)$ can uniformly converge to a bounded neighborhood of the optimal performance index function.

For convenience of analysis, we transform the expressions of approximation errors. According to (27), we have $\hat{V}_i(x_k) \leq \Gamma_i(x_k) + \epsilon$. Then, for $\forall i = 0, 1, \ldots$, there exists a finite constant $\sigma \geq 1$ that makes

$$\hat{V}_i(x_k) \leq \sigma\Gamma_i(x_k) \tag{29}$$

hold uniformly. Hence, we can give the following theorem.

Theorem 3.2: Let $x_k \in \mathbb{R}^n$ be an arbitrary controllable state and Assumptions 1–4 hold. For $\forall\, i = 0, 1, \ldots$, let $\Gamma_i(x_k)$ be expressed as (17) and $\hat{V}_i(x_k)$ be expressed as (16). Let $0 < \gamma < \infty$ and $1 \leq \delta < \infty$ be both constant that make

$$J^*\left(F(x_k, u_k)\right) \leq \gamma U(x_k, u_k) \tag{30}$$

$$J^*(x_k) \leq V_0(x_k) \leq \delta J^*(x_k) \tag{31}$$

hold uniformly. If there exists $1 \leq \sigma < \infty$ that makes (29) hold uniformly, then we have

$$\hat{V}_i(x_k) \leq \sigma \left(1+\sum_{j=1}^{i} \frac{\gamma^j \sigma^{j-1}(\sigma-1)}{(\gamma+1)^j}+\frac{\gamma^i \sigma^i(\delta-1)}{(\gamma+1)^i}\right) J^*(x_k) \tag{32}$$

where we define $\sum_j^i(\cdot)=0$, for $\forall j > i$ and $i, j = 0, 1, \ldots$.

Proof: The theorem can be proved by mathematical induction. First, let $i = 0$. Then, (32) becomes

$$\hat{V}_0(x_k) \leq \sigma\delta J^*(x_k). \tag{33}$$

As $V_0(x_k) = \Gamma_0(x_k) = \theta\Psi(x_k)$, we can obtain $\sigma\Gamma_0(x_k) \leq \sigma\delta J^*(x_k)$. Therefore, the conclusion holds for $i = 0$.

Next, let $i = 1$. According to (17), we have

$$\begin{aligned}
\Gamma_1(x_k) &= \min_{u_k\in\mathbb{R}^m} \left\{U(x_k,u_k)+\hat{V}_0\left(F(x_k,u_k)\right)\right\} \\
&\leq \min_{u_k\in\mathbb{R}^m} \left\{U(x_k,u_k)+\sigma\delta J^*\left(F(x_k,u_k)\right)\right\} \\
&\leq \min_{u_k\in\mathbb{R}^m} \left\{\left(1+\gamma\frac{\sigma\delta-1}{\gamma+1}\right)U(x_k,u_k)\right. \\
&\qquad \left.+\left(\sigma\delta-\frac{\sigma\delta-1}{\gamma+1}\right)J^*\left(F(x_k,u_k)\right)\right\} \\
&= \left(1+\gamma\frac{\sigma\delta-1}{\gamma+1}\right) \\
&\quad \times \min_{u_k\in\mathbb{R}^m} \left\{U(x_k,u_k)+J^*\left(F(x_k,u_k)\right)\right\} \\
&= \left(1+\frac{\gamma(\sigma-1)}{\gamma+1}+\frac{\gamma\sigma(\delta-1)}{\gamma+1}\right)J^*(x_k).
\end{aligned} \tag{34}$$

According to (29), we can obtain

$$\hat{V}_1(x_k) \leq \sigma\left(1+\frac{\gamma(\sigma-1)}{\gamma+1}+\frac{\gamma\sigma(\delta-1)}{\gamma+1}\right)J^*(x_k) \tag{35}$$

which shows that (32) holds for $i = 1$.

Assume that (32) holds for $i = l-1$, where $l = 1, 2, \ldots$. Then, for $i = l$, we have (36), shown at the bottom of the page.

Then, according to (29), we can obtain (32), which proves the conclusion for $\forall\, i = 0, 1, \ldots$. ■

From (32), we can see that for arbitrary finite i, σ, and δ, there exists a bounded error between the iterative performance index function $\hat{V}_i(x_k)$ and the optimal performance index function $J^*(x_k)$. As $i \to \infty$, the bound of the approximation errors may increase to infinity. Thus, in the following, we will show the convergence properties of the iterative ADP algorithm (13)–(16) using the error bound method.

Theorem 3.3: Let $x_k \in \mathbb{R}^n$ be an arbitrary controllable state and Assumptions 1–4 hold. Suppose Theorem 3.2 holds for $\forall x_k \in \mathbb{R}^n$. If for $0 < \gamma < \infty$ the inequality

$$1 \leq \sigma < \frac{\gamma+1}{\gamma} \tag{37}$$

holds, then as $i \to \infty$, the iterative performance index function $\hat{V}_i(x_k)$ in the iterative ADP algorithm (13)–(16) is uniformly convergent to a bounded neighborhood of the optimal performance index function $J^*(x_k)$, i.e.,

$$\lim_{i\to\infty} \hat{V}_i(x_k) = \hat{V}_\infty(x_k) \leq \sigma\left(1+\frac{\gamma(\sigma-1)}{1-\gamma(\sigma-1)}\right)J^*(x_k). \tag{38}$$

$$\begin{aligned}
\Gamma_l(x_k) &= \min_{u_k\in\mathbb{R}^m} \left\{U(x_k,u_k)+\hat{V}_{l-1}\left(F(x_k,u_k)\right)\right\} \\
&\leq \min_{u_k\in\mathbb{R}^m} \left\{U(x_k,u_k)+\sigma\left(1+\sum_{j=1}^{l-1}\frac{\gamma^j\sigma^{j-1}(\sigma-1)}{(\gamma+1)^j}+\frac{\gamma^{l-1}\sigma^{l-1}(\delta-1)}{(\gamma+1)^{l-1}}\right)J^*(x_{k+1})\right\} \\
&\leq \min_{u_k\in\mathbb{R}^m} \left\{\left(1+\gamma\sum_{j=1}^{l-1}\frac{\gamma^{j-1}\sigma^{j-1}(\sigma-1)}{(\gamma+1)^j}+\frac{\gamma^{l-1}\sigma^{l-1}(\sigma\delta-1)}{(\gamma+1)^{l-1}}\right)U(x_k,u_k)\right. \\
&\qquad +\left[\sigma\left(1+\sum_{j=1}^{l-1}\frac{\gamma^j\sigma^{j-1}(\sigma-1)}{(\gamma+1)^j}+\frac{\gamma^{l-1}\sigma^{l-1}(\delta-1)}{(\gamma+1)^{l-1}}\right)\right. \\
&\qquad \left.\left.-\left(\sum_{j=1}^{l-1}\frac{\gamma^{j-1}\sigma^{j-1}(\sigma-1)}{(\gamma+1)^j}+\frac{\gamma^{l-1}\sigma^{l-1}(\sigma\delta-1)}{(\gamma+1)^{l-1}}\right)\right]J^*\left(F(x_k,u_k)\right)\right\} \\
&= \left(1+\sum_{j=1}^{l}\frac{\gamma^j\sigma^{j-1}(\sigma-1)}{(\gamma+1)^j}+\frac{\gamma^l\sigma^l(\delta-1)}{(\gamma+1)^l}\right)\min_{u_k\in\mathbb{R}^m}\left\{U(x_k,u_k)+J^*\left(F(x_k,u_k)\right)\right\} \\
&= \left(1+\sum_{j=1}^{l}\frac{\gamma^j\sigma^{j-1}(\sigma-1)}{(\gamma+1)^j}+\frac{\gamma^l\sigma^l(\delta-1)}{(\gamma+1)^l}\right)J^*(x_k)
\end{aligned} \tag{36}$$

Proof: According to (36) in Theorem 3.2, we can see that for $j = 1, 2, \ldots$, the sequence $\{\gamma^j \sigma^{j-1}(\sigma-1)/(\gamma+1)^j\}$ is a geometric series. Then, (36) can be written as

$$\Gamma_i(x_k) \leq \left(1 + \frac{\frac{\gamma(\sigma-1)}{\gamma+1}\left(1-\left(\frac{\gamma\sigma}{\gamma+1}\right)^i\right)}{1-\frac{\gamma\sigma}{\gamma+1}} + \frac{\gamma^i\sigma^i(\delta-1)}{(\gamma+1)^i}\right) J^*(x_k). \tag{39}$$

As $i \to \infty$, if $1 \leq \sigma < (\gamma+1)/\gamma$, then (39) becomes

$$\lim_{i\to\infty} \Gamma_i(x_k) = \Gamma_\infty(x_k) \leq \left(1 + \frac{\gamma(\sigma-1)}{1-\gamma(\sigma-1)}\right) J^*(x_k). \tag{40}$$

According to (29), let $i \to \infty$, we have

$$\hat{V}_\infty(x_k) \leq \sigma\Gamma_\infty(x_k). \tag{41}$$

According to (40) and (41), we can obtain (38). ■

Corollary 3.1: Let $x_k \in \mathbb{R}^n$ be an arbitrary controllable state and Assumptions 1–4 hold. Suppose Theorem 3.2 holds for $\forall x_k \in \mathbb{R}^n$. If for $0 < \gamma < \infty$, $1 \leq \sigma < \infty$, and the inequality (37) holds, then the control law $\hat{v}_i(x_k)$ of the iterative ADP algorithm (13)–(16) is convergent, i.e.,

$$\hat{v}_\infty(x_k) = \lim_{i\to\infty} \hat{v}_i(x_k). \tag{42}$$

Remark 3.3: Theorem 3.3 gives a convergent condition for the iterative performance index function $\hat{V}_i(x_k)$ in the iterative ADP algorithm (13)–(16). From the convergent condition (37), we can see that the approximation error σ is only the function of variable γ, which is independent of δ. As $i \to \infty$, the limit of the iterative performance index function in (38) is the function of variables σ and γ while it is still independent of δ. This makes the justification of the convergence and estimation of the bound of the performance index function easier.

Due to the existence of approximation errors, the iterative performance index function $\hat{V}_i(x_k)$ is not necessary larger or smaller than $\Gamma_i(x_k)$. If (29) is only satisfied for some $0 < \sigma' < 1$, there must exist some $\sigma \geq 1$ that also satisfies (29) for different x_k and i. From (29), we can see that $0 < \sigma' < 1$ is just a special case of $\sigma \geq 1$. Therefore, in this paper, we will consider only the situation for $\sigma \geq 1$.

For the iterative ADP algorithm with finite approximation errors, we can see that, for different approximation errors, the limits of the iterative performance index function $\hat{V}_i(x_k)$ are different. This property can be proved by the following theorem.

Theorem 3.4: Let $x_k \in \mathbb{R}^n$ be an arbitrary controllable state and Assumptions 1–4 hold. Let

$$\bar{V}_\infty(x_k) = \sigma\left(1 + \frac{\gamma(\sigma-1)}{1-\gamma(\sigma-1)}\right) J^*(x_k) \tag{43}$$

be the least upper bound of the limit of the iterative performance index function. If Theorem 3.3 holds for $\forall x_k \in \mathbb{R}^n$, then we have $\bar{V}_\infty(x_k)$ as a monotonically increasing function of the approximation error coefficient σ.

Proof: From (43), we can see that the least upper bound of the limit of the iterative performance index function $\bar{V}_\infty(x_k)$ is a differentiable function of the approximation error coefficient σ. Hence, we can take the derivative of the approximation error σ along to the two sides of (43). Then, we can obtain

$$\frac{\partial\bar{V}_\infty(x_k)}{\partial\sigma} = \left(\frac{1+\gamma}{(\gamma(\sigma-1)-1)^2}\right) J^*(x_k) > 0. \tag{44}$$

This proves the theorem. ■

According to the definitions of iterative performance index functions $\hat{V}_i(x_k)$ and $\Gamma_i(x_k)$ in (16) and (17), for $\forall\, i = 0, 1, \ldots$, if we let

$$\hat{V}_i(x_k) - \Gamma_i(x_k) = \epsilon_i(x_k) \tag{45}$$

then we can define

$$\epsilon = \sup\{\epsilon_0(x_k), \epsilon_1(x_k), \ldots\}. \tag{46}$$

In the previous section, the approximation error σ that satisfies (29) is used to analyze the convergence properties of the iterative ADP algorithm. Generally, the approximation error ϵ is obtained instead of obtaining σ. Therefore, if we use ϵ to express the approximation error, then an expression transformation is needed between the two approximation errors ϵ and σ.

For $\forall\, i = 0, 1, \ldots$, for any $\epsilon_i(x_k)$ expressed in (45), there exists a $\sigma_i(x_k)$ that satisfies

$$\hat{V}_i(x_k) - \epsilon_i(x_k) = \frac{\hat{V}_i(x_k)}{\sigma_i(x_k)}. \tag{47}$$

According to (46), we can also obtain $\sigma = \sup\{\sigma_0(x_k), \sigma_1(x_k), \ldots\}$.

Remark 3.4: In [6], the approximation error for a class of single-layer neural networks is discussed to show the uniform convergence of a performance index function, whereas for different types of neural networks, the convergence cannot be guaranteed. In this paper, we can see that the type of errors in the algorithm is not specified. For example, the errors can denote the approximation errors by arbitrary neural networks and fuzzy structures. Therefore, the approximation errors discussed in this paper are more general than the one in [6].

IV. Neural Network Implementation for the Optimal Control Scheme

In the case of linear systems, the performance index function is quadratic and the control law is linear. In the nonlinear case, this is not necessarily true, and therefore, we use back propagation (BP) neural networks to approximate $v_i(x_k)$ and $V_i(x_k)$.

Assume that the number of hidden layer neurons is denoted by L. The weight matrix between the input and hidden layers is denoted by Y. The weight matrix between the hidden and output layers is denoted by W. The input vector of the neural network is denoted as X. Then, the output of the three-layer neural network is represented by

$$\hat{F}(X, Y, W) = W\sigma(YX) \tag{48}$$

where $\sigma(YX) \in R^L$, $[\sigma(z)]_i = (e^{z_i} - e^{-z_i})/(e^{z_i} + e^{-z_i})$, $i = 1, \ldots L$, are the activation functions.

The neural network estimation error can be expressed by

$$F(X) = F(X, Y^*, W^*) + \varepsilon(X) \tag{49}$$

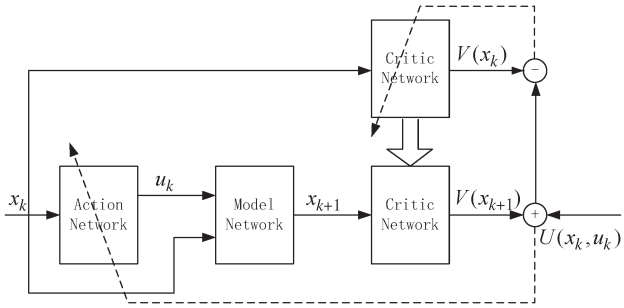

Fig. 1. Structure diagram of the algorithm.

where Y^* and W^* are the ideal weight parameters, and $\varepsilon(X)$ is the reconstruction error.

Here, there are two networks, which are the critic and action networks, respectively. Both neural networks are chosen as three-layer feedforward networks. The whole structure diagram is shown in Fig. 1.

A. Critic Network

The critic network is used to approximate the performance index function $V_i(x_k)$. The output of the critic network is denoted by

$$\hat{V}_{i+1}(x_k) = W_{ci}(k)\sigma\left(Y_{ci}(k)x_k\right). \tag{50}$$

The target function can be written as

$$V_{i+1}(x_k) = U\left(x_k, v_i(x_k)\right) + \hat{V}_i(x_{k+1}). \tag{51}$$

Then, we define the error function for the critic network as

$$e_{ci}(k) = \hat{V}_{i+1}(x_k) - V_{i+1}(x_k). \tag{52}$$

The objective function to be minimized for the critic network training is

$$E_{ci}(k) = \frac{1}{2}e_{ci}^2(k). \tag{53}$$

The gradient-based weight update rule [24] that can be applied here to the critic network training is given by

$$w_{c(i+1)}(k) = w_{ci}(k) + \Delta w_{ci}(k) \tag{54}$$

$$\Delta w_{ci}(k) = -\alpha_c\left[\frac{\partial E_{ci}(k)}{\partial w_{ci}(k)}\right] \tag{55}$$

$$\frac{\partial E_{ci}(k)}{\partial w_{ci}(k)} = \frac{\partial E_{ci}(k)}{\partial \hat{V}_{i+1}(x_k)}\frac{\partial \hat{V}_{i+1}(x_k)}{\partial w_{ci}(k)} \tag{56}$$

where $\alpha_c > 0$ is the learning rate of the critic network, and $w_{ci}(k)$ is the weight vector in the critic network, which can be replaced by W_{ci} and Y_{ci}, respectively.

If we define

$$q_l(k) = \sum_{j=1}^{L_c} Y_{ci}^{l_j}(k)x_{jk}, \quad l = 1, 2\ldots, L_c \tag{57}$$

$$p_l(k) = \frac{e^{q_l(k)} - e^{-q_l(k)}}{e^{q_l(k)} + e^{-q_l(k)}}, \quad l = 1, 2\ldots, L_c \tag{58}$$

then we have

$$\hat{V}_{i+1}(x_k) = \sum_{l=1}^{L_c} W_{ci}^l(k)p_l(k) \tag{59}$$

where L_c is the total number of hidden nodes in the critic network. By applying the chain rule, the adaptation of the critic network is summarized as follows.

The hidden-to-output layer of the critic is updated as

$$\begin{aligned}\Delta W_{ci}^l(k) &= -\alpha_c\frac{\partial E_{ci}(k)}{\partial \hat{V}_{i+1}(x_k)}\frac{\partial \hat{V}_{i+1}(x_k)}{\partial W_{ci}^l(k)} \\ &= -\alpha_c e_{ci}(k)p_l(k).\end{aligned} \tag{60}$$

The input-to-hidden layer of the critic is updated as

$$\begin{aligned}\Delta Y_{ci}^l(k) &= -\alpha_c\frac{\partial E_{ci}(k)}{\partial \hat{V}_{i+1}(x_k)}\frac{\partial \hat{V}_{i+1}(x_k)}{\partial Y_{ci}^l(k)} \\ &= -\alpha_c\frac{\partial E_{ci}(k)}{\partial \hat{V}_{i+1}(x_k)}\frac{\partial \hat{V}_{i+1}(x_k)}{\partial p_l(k)}\frac{\partial p_l(k)}{\partial q_l(k)}\frac{\partial q_l(k)}{\partial Y_{ci}^l(k)} \\ &= -\alpha_c e_{ci}(k)W_{ci}^l(k)\left[\frac{1}{2}\left(1 - p_l^2(t)\right)\right]x_{lk}.\end{aligned} \tag{61}$$

B. Action Network

In the action network, the state error x_k is used as an input to create the optimal control law as the output of the network. The output can be formulated as

$$\hat{v}_i(x_k) = W_{ai}\sigma(Y_{ai}x_k). \tag{62}$$

The target of the output of the action network is given by (11). Therefore, we can define the output error of the action network as

$$e_{ai}(k) = \hat{v}_i(x_k) - v_i(x_k) \tag{63}$$

where the target function of the iterative control law is given by

$$v_i(x_k) = \arg\min_{u_k}\left\{U(x_k, u_k) + \hat{V}_{i-1}(x_{k+1})\right\}. \tag{64}$$

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

$$E_{ai}(k) = \frac{1}{2}e_{ai}^2(k). \tag{65}$$

The weights updating algorithm is similar to the one for the critic network. By the gradient descent rule, we can obtain

$$w_{a(i+1)}(k) = w_{ai}(k) + \Delta w_{ai}(k) \tag{66}$$

$$\Delta w_{ai}(k) = -\beta_a\left[\frac{\partial E_{ai}(k)}{\partial w_{ai}(k)}\right] \tag{67}$$

$$\frac{\partial E_{ai}(k)}{\partial w_{ai}(k)} = \frac{\partial E_{ai}(k)}{\partial e_{ai}(k)}\frac{\partial e_{ai}(k)}{\partial \hat{v}_i(x_k)}\frac{\partial \hat{v}_i(x_k)}{\partial w_{ai}(k)} \tag{68}$$

where $\beta_a > 0$ is the learning rate of the action network, and $w_{ai}(k)$ is the weight of the action network, which can be replaced by W_{ai} and Y_{ai}.

If we define

$$g_l(k) = \sum_{j=1}^{L_a} Y_{ai}^{l_j}(k) x_{jk}, \quad l = 1, 2 \ldots, L_a \tag{69}$$

$$h_l(k) = \frac{e^{g_l(k)} - e^{-g_l(k)}}{e^{g_l(k)} + e^{-g_l(k)}}, \quad l = 1, 2 \ldots, L_a \tag{70}$$

then we have

$$\hat{v}_i(x_k) = \sum_{l=1}^{L_a} W_{ai}^{l}(k) h_l(k) \tag{71}$$

where L_a is the total number of hidden nodes in the action network. By applying the chain rule, the adaptation of the action network is summarized as follows.

The hidden-to-output layer of the action is updated as

$$\begin{aligned} \Delta W_{ai}^{l}(k) &= -\beta_a \frac{\partial E_{ai}(k)}{\partial \hat{v}_i(x_k)} \frac{\partial \hat{v}_i(x_k)}{\partial W_{ai}^{l}(k)} \\ &= -\beta_a e_{ai}(k) h_l(k). \end{aligned} \tag{72}$$

The input-to-hidden layer of the action is updated as

$$\begin{aligned} \Delta Y_{ai}^{l}(k) &= -\beta_a \frac{\partial E_{ai}(k)}{\partial \hat{v}_i(x_k)} \frac{\partial \hat{v}_i(x_k)}{\partial Y_{ai}^{l}(k)} \\ &= -\beta_a \frac{\partial E_{ai}(k)}{\partial \hat{v}_i(x_k)} \frac{\partial \hat{v}_i(x_k)}{\partial h_l(k)} \frac{\partial h_l(k)}{\partial g_l(k)} \frac{\partial g_l(k)}{\partial Y_{ai}^{l}(k)} \\ &= -\beta_a e_{ai}(k) W_{ai}^{l}(k) \left[\frac{1}{2}\left(1 - g_l^2(t)\right)\right] x_{lk}. \end{aligned} \tag{73}$$

Details on neural training algorithm can be also found in [24].

Remark 4.1: Using neural networks, we can implement the proposed iterative ADP algorithm effectively. As neural networks are used, it should be pointed out that the neural-networked-based optimization is generally related to local optimum. To overcome this problem, many improved training algorithms are proposed to achieve the global optimum. Decreasing the learning rate is an effective method to achieve the global optimum, and it is also recommended to initialize the weights of the neural network to small values to obtain the global approximation results effectively [10]. Many other improved training algorithms are proposed to achieve the global optimum [9], [10], such as momentum algorithm, variable learning rate algorithm, and Levenberg–Marquardt algorithm. In this paper, we adopt small learning rates for training the critic and action networks to achieve the global optimal solution.

V. Simulation Studies

To evaluate the performance of our iterative ADP algorithm with finite approximation errors, we choose two examples with quadratic utility functions for numerical experiments.

Example 5.1: Our first example is chosen as the example in [11] and [27]. Consider the following discrete-time nonaffine nonlinear system:

$$x_{k+1} = F(x_k, u_k) = x_k + \sin\left(0.1x_k^2 + u_k\right). \tag{74}$$

The initial state is $x_0 = 1$. Since $F(0, 0) = 0$, $x_k = 0$ is an equilibrium state of system (74). However, since $(\partial F(x_k, u_k)/\partial x_k)(0, 0) = 1$, nonlinear system (74) is marginally stable at $x_k = 0$ and the equilibrium $x_k = 0$ is not attractive. Let the performance index function be in quadratic form, which is expressed as

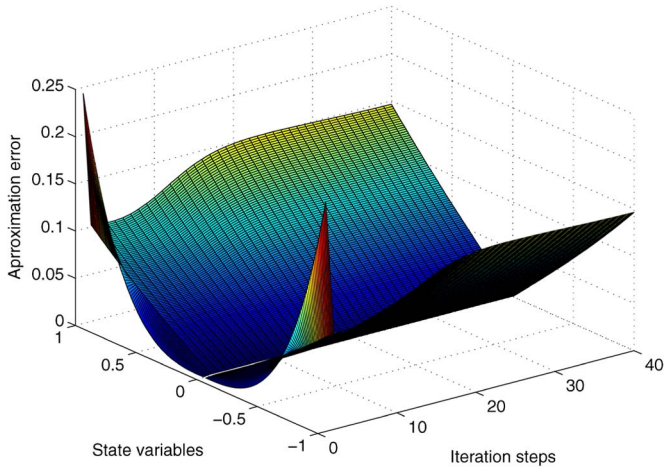

Fig. 2. Curve of the approximation errors.

$$J(x_0, \underline{u}_0) = \sum_{k=0}^{\infty} \left(x_k^T Q x_k + u_k^T R u_k\right) \tag{75}$$

where matrix $Q = R = I$, and I denotes the identity matrix with suitable dimensions.

Neural networks are used to implement the iterative ADP algorithm. The critic and action networks are chosen as three-layer BP neural networks with the structures of 1–8–1 and 1–8–1, respectively. Neural networks are used to implement the iterative ADP algorithm, and the neural network structure diagram of the algorithm can be also seen in [24], [28], and [36]. Let $\theta = 8$ and $\Psi(x_k) = x_k^T Q x_k$ to initialize the algorithm.

The curve of the approximation errors defined in (45) is displayed in Fig. 2.

We choose a random array of state variable in $[-1, 1]$ to train the neural networks. For each iterative step, the critic and action networks are trained for 1000 steps under the learning rate $\alpha = 0.001$ so that the approximation error is reached. The iterative ADP algorithm runs for 35 iteration steps to guarantee the convergence of the iterative performance index function. To show the effectiveness of the proposed iterative ADP algorithm, we choose four different global training precision values of neural networks. The approximation errors of the neural networks are chosen as $\epsilon = 10^{-6}$, $\epsilon = 10^{-4}$, $\epsilon = 10^{-3}$, and $\epsilon = 10^{-1}$. The trajectory of the iterative performance index function is shown in Fig. 3(a)–(d) for the approximation errors of the neural networks $\epsilon = 10^{-6}$, $\epsilon = 10^{-4}$, $\epsilon = 10^{-3}$, and $\epsilon = 10^{-1}$, respectively.

For approximation error $\epsilon = 10^{-6}$, implement the approximate optimal control for system (74). Let the implementation time $T_f = 20$, and the trajectories of the control and state are displayed in Fig. 4(a). For approximation error $\epsilon = 10^{-4}$, and the trajectories of the control and state are displayed in Fig. 4(b). When the approximation error $\epsilon = 10^{-3}$, we can see that the iterative performance index functions are not monotone. The trajectories of the control and state are displayed in

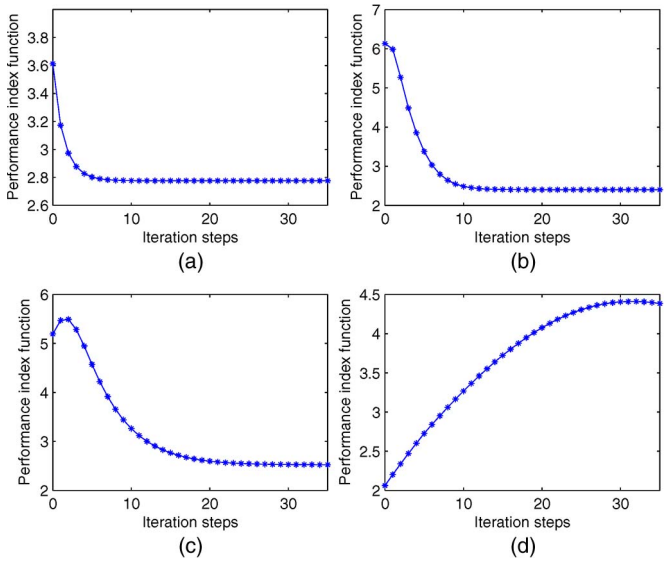

Fig. 3. Trajectories of the iterative performance index functions. (a) Approximation error $\epsilon = 10^{-6}$. (b) Approximation error $\epsilon = 10^{-4}$. (c) Approximation error $\epsilon = 10^{-3}$. (d) Approximation error $\epsilon = 10^{-1}$.

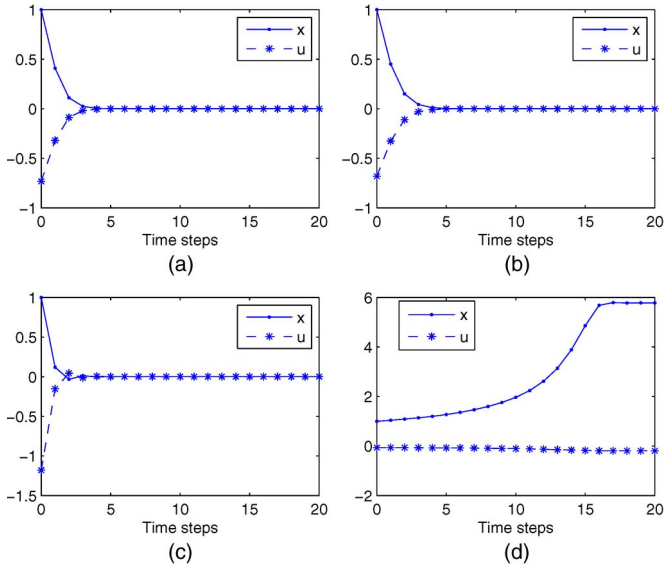

Fig. 4. Control and state trajectories. (a) Approximation error $\epsilon = 10^{-6}$. (b) Approximation error $\epsilon = 10^{-4}$. (b) Approximation error $\epsilon = 10^{-3}$. (d) Approximation error $\epsilon = 10^{-1}$.

Fig. 4(c). When the approximation error $\epsilon = 10^{-1}$, we can see that the iterative performance index functions are not convergent. In this situation, the control system is not stable and the trajectories of the control and state are displayed in Fig. 4(d).

Example 5.2: The second example is chosen as the example in [21] and [25]. We consider the following system:

$$\begin{aligned} x_1(k+1) &= \left[x_1^2(k) + x_2^2(k) + u(k)\right] \cos\left(x_2(k)\right) \\ x_2(k+1) &= \left[x_1^2(k) + x_2^2(k) + u(k)\right] \sin\left(x_2(k)\right). \end{aligned} \quad (76)$$

Let $x_k = [x_1(k), x_2(k)]^T$ denote the system state vector and $u_k = u(k)$ denote the control. The performance index function is defined as in (75) of Example 1.

The initial state is $x_0 = [1, -1]^T$. From system (76), we can see that $x_k = 0$ is an equilibrium state and the autonomous system $F(x_k, 0)$ is unstable. We also use neural networks to

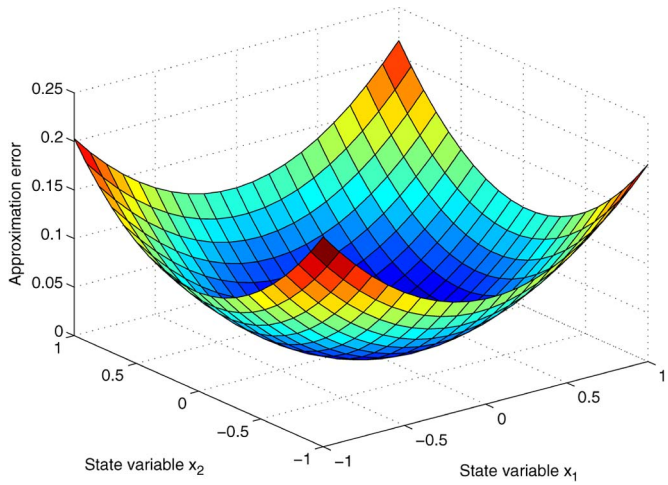

Fig. 5. Curve of the approximation errors.

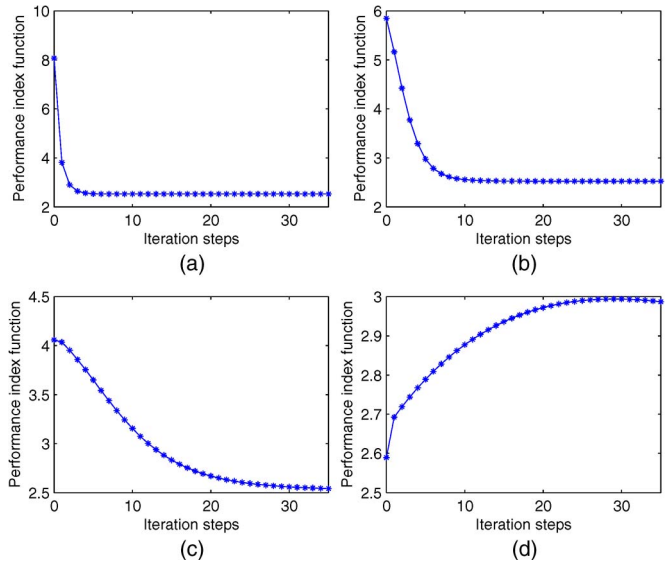

Fig. 6. Trajectories of the iterative performance index functions. (a) Approximation error $\epsilon = 10^{-8}$. (b) Approximation error $\epsilon = 10^{-4}$. (c) Approximation error $\epsilon = 10^{-3}$. (d) Approximation error $\epsilon = 10^{-1}$.

implement the iterative ADP algorithm. The critic and action networks are chosen as three-layer BP neural networks with the structures of 2–8–1 and 2–8–1, respectively. We also use the critic network to approximate the iterative performance index functions and use the action network to approximate the iterative control laws.

Let $\theta = 8$ and $\Psi(x_k) = x_k^T Q x_k$ to initialize the algorithm. The iterative ADP algorithm runs for 35 iteration steps to guarantee the convergence of the iterative performance index function. The curve of the converged approximation errors is displayed in Fig. 5.

To show the effectiveness of the proposed iterative ADP algorithm, we also choose four different approximation precision values of neural networks. For each iterative step, the critic and action networks are trained for 1500 steps under the learning rate $\alpha = 0.001$ so that the approximation error is reached. First, let the approximation errors of the neural networks be $\epsilon = 10^{-8}$, $\epsilon = 10^{-4}$, $\epsilon = 10^{-3}$, and $\epsilon = 10^{-1}$. The trajectories of the iterative performance index function are shown in Fig. 6(a)–(d), respectively.

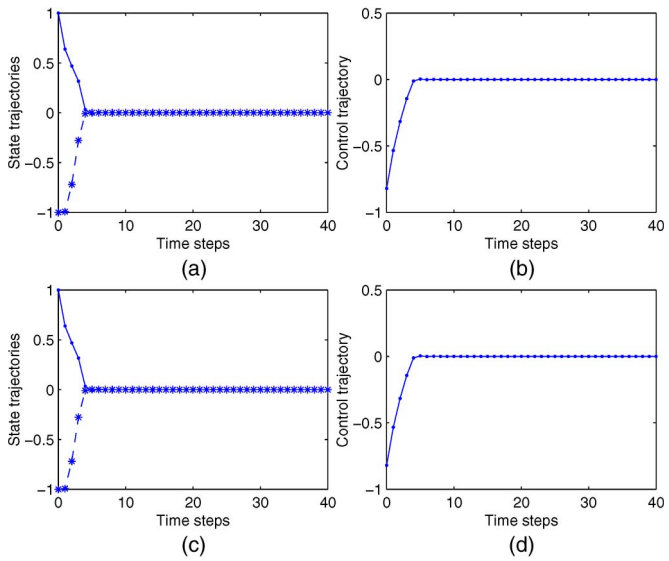

Fig. 7. Control and state trajectories. (a) State trajectories for approximation error $\epsilon = 10^{-8}$. (b) Control trajectory for approximation error $\epsilon = 10^{-8}$. (c) State trajectories for approximation error $\epsilon = 10^{-4}$. (d) Control trajectory for approximation error $\epsilon = 10^{-4}$.

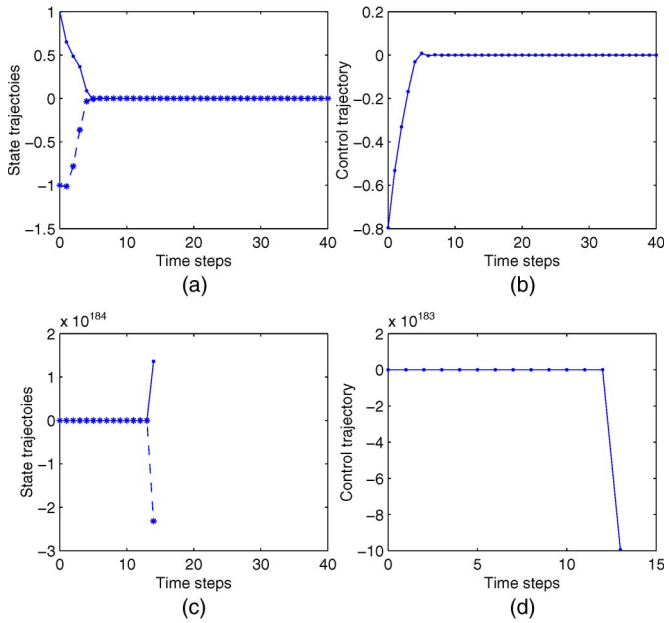

Fig. 8. Control and state trajectories. (a) State trajectories for approximation error $\epsilon = 10^{-3}$. (b) Control trajectory for approximation error $\epsilon = 10^{-3}$. (c) State trajectories for approximation error $\epsilon = 10^{-1}$. (d) Control trajectory for approximation error $\epsilon = 10^{-1}$.

For approximation error $\epsilon = 10^{-8}$, implement the approximate optimal control for system (76). Let the implementation time $T_f = 40$. The trajectories of the state are displayed in Fig. 7(a), and the corresponding control trajectory is displayed in Fig. 7(b). For approximation error $\epsilon = 10^{-4}$, the trajectories of the state are displayed in Fig. 7(c) and the corresponding control trajectory is displayed in Fig. 7(d). When the approximation error $\epsilon = 10^{-3}$, we can see that the iterative performance index functions are not completely converged within 35 iteration steps. The trajectories of the state are displayed in Fig. 8(a), and the corresponding control trajectory is displayed in Fig. 8(b). When the approximation error $\epsilon = 10^{-1}$, we can see that the iterative performance index functions are not convergent. The control system is not stable. The trajectories of the state are displayed in Fig. 8(c), and the corresponding control trajectory is displayed in Fig. 8(d).

VI. Conclusion

In this paper, an effective iterative ADP algorithm has been proposed to solve optimal control problems for infinite-horizon discrete-time nonlinear systems. When the iterative control law and the iterative performance index function in each iteration cannot be accurately obtained, it has been shown that the iterative performance index functions can converge to the finite neighborhood of the optimal performance index function if the convergence conditions are satisfied. Neural networks are used to implement the iterative ADP algorithm. Finally, two simulation examples are given to illustrate the performance of the proposed algorithm.

References

[1] M. Abu-Khalaf and F. L. Lewis, "Nearly optimal control laws for nonlinear systems with saturating actuators using a neural network HJB approach," *Automatica*, vol. 41, no. 5, pp. 779–791, May 2005.

[2] A. Al-Tamimi, F. L. Lewis, and M. Abu-Khalaf, "Discrete-time nonlinear HJB solution using approximate dynamic programming: Convergence proof," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 38, no. 4, pp. 943–949, Aug. 2008.

[3] S. N. Balakrishnan and V. Biega, "Adaptive-critic-based neural networks for aircraft optimal control," *J. Guid., Control, Dyn.*, vol. 19, no. 4, pp. 893–898, Jul./Aug. 1996.

[4] R. Beard, "Improving the closed-loop performance of nonlinear systems," Ph.D. dissertation, Rensselaer Polytechnic Inst., Troy, NY, 1995.

[5] D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-Dynamic Programming*. Cambridge, MA: Athena Scientific, 1996.

[6] T. Cheng, F. L. Lewis, and M. Abu-Khalaf, "A neural network solution for fixed-final time optimal control of nonlinear systems," *Automatica*, vol. 43, no. 3, pp. 482–490, Mar. 2007.

[7] R. Enns and J. Si, "Helicopter trimming and tracking control using direct neural dynamic programming," *IEEE Trans. Neural Netw.*, vol. 14, no. 4, pp. 929–939, Aug. 2003.

[8] S. Ferrari, J. E. Steck, and R. Chandramohan, "Adaptive feedback control by constrained approximate dynamic programming," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 38, no. 4, pp. 982–987, Aug. 2008.

[9] M. T. Hagan, H. B. Demuth, and M. H. Beale, *Neural Network Design*. Boston, MA: PWS-Kent, 1996.

[10] S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd ed. Upper Saddle River, NJ: Prentice-Hall, 1999.

[11] N. Jin, D. Liu, T. Huang, and Z. Pang, "Discrete-time adaptive dynamic programming using wavelet basis function neural networks," in *Proc. IEEE Symp. Approx. Dyn. Programm. Reinforcement Learn.*, Honolulu, HI, USA, Apr. 2007, pp. 135–142.

[12] R. V. Kulkarni and G. K. Venayagamoorthy, "Bio-inspired algorithms for autonomous deployment and localization of sensor nodes," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 40, no. 6, pp. 663–675, Nov. 2010.

[13] G. G. Lendaris, "Higher level application of ADP: A next phase for the control field?" *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 38, no. 4, pp. 901–912, Aug. 2008.

[14] F. L. Lewis and V. G. Kyriakos, "Reinforcement learning for partially observable dynamic processes: Adaptive dynamic programming using measured output data," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 41, no. 1, pp. 14–25, Jan. 2011.

[15] F. L. Lewis and D. Vrabie, "Reinforcement learning and adaptive dynamic programming for feedback control," *IEEE Circuits Syst. Mag.*, vol. 9, no. 3, pp. 32–50, Third Quarter, 2009.

[16] B. Lincoln and A. Rantzer, "Relaxing dynamic programming," *IEEE Trans. Autom. Control*, vol. 51, no. 8, pp. 1249–1260, Aug. 2006.

[17] D. Liu, H. Javaherian, O. Kovalenko, and T. Huang, "Adaptive critic learning techniques for engine torque and air-fuel ratio control," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 38, no. 4, pp. 988–993, Aug. 2008.
[18] D. Liu, Y. Zhang, and H. Zhang, "A self-learning call admission control scheme for CDMA cellular networks," *IEEE Trans. Neural Netw.*, vol. 16, no. 5, pp. 1219–1228, Sep. 2005.
[19] J. Mao and C. G. Cassandras, "Optimal control of multi-stage discrete event systems with real-time constraints," *IEEE Trans. Autom. Control*, vol. 54, no. 1, pp. 108–123, Jan. 2009.
[20] J. J. Murray, C. J. Cox, G. G. Lendaris, and R. Saeks, "Adaptive dynamic programming," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 32, no. 2, pp. 140–153, May 2002.
[21] E. M. Navarro-Lopez, "Local feedback passivation of nonlinear discrete-time systems through the speed-gradient algorithm," *Automatica*, vol. 43, no. 7, pp. 1302–1306, Jul. 2007.
[22] D. V. Prokhorov and D. C. Wunsch, "Adaptive critic designs," *IEEE Trans. Neural Netw.*, vol. 8, no. 5, pp. 997–1007, Sep. 1997.
[23] J. Seiffertt, S. Sanyal, and D. C. Wunsch, "Hamilton–Jacobi–Bellman equations and approximate dynamic programming on time scales," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 38, no. 4, pp. 918–923, Aug. 2008.
[24] J. Si and Y.-T. Wang, "On-line learning control by association and reinforcement," *IEEE Trans. Neural Netw.*, vol. 12, no. 2, pp. 264–276, Mar. 2001.
[25] H. Sira-Ramirez, "Non-linear discrete variable structure systems in quasi-sliding mode," *Int. J. Control*, vol. 54, no. 5, pp. 1171–1187, May 1991.
[26] K. G. Vamvoudakis and F. L. Lewis, "Multi-player non-zero-sum games: Online adaptive learning solution of coupled Hamilton–Jacobi equations," *Automatica*, vol. 47, no. 8, pp. 1556–1569, Aug. 2011.
[27] F. Wang, N. Jin, D. Liu, and Q. Wei, "Adaptive dynamic programming for finite-horizon optimal control of discrete-time nonlinear systems with ϵ-error bound," *IEEE Trans. Neural Netw.*, vol. 22, no. 1, pp. 24–36, Jan. 2011.
[28] F. Wang, H. Zhang, and D. Liu, "Adaptive dynamic programming: An introduction," *IEEE Comput. Intell. Mag.*, vol. 4, no. 2, pp. 39–47, May 2009.
[29] C. Watkins, "Learning from delayed rewards," Ph.D. dissertation, Cambridge Univ., Cambridge, U.K., 1989.
[30] Q. Wei and D. Liu, "Adaptive dynamic programming with stable value iteration algorithm for discrete-time nonlinear systems," in *Proc. Int. Joint Conf. Neural Netw.*, Brisbane, Australia, Jun. 2012, pp. 1–6.
[31] Q. Wei and D. Liu, "A novel optimal control scheme for discrete-time nonlinear systems using iterative adaptive dynamic programming," *IEEE Trans. Autom. Sci. Eng.*, submitted for publication.
[32] Q. Wei, H. Zhang, and J. Dai, "Model-free multiobjective approximate dynamic programming for discrete-time nonlinear systems with general performance index functions," *Neurocomputing*, vol. 72, no. 7–9, pp. 1839–1848, Mar. 2009.
[33] P. J. Werbos, "A menu of designs for reinforcement learning over time," in *Neural Networks for Control*, W. T. Miller, R. S. Sutton, and P. J. Werbos, Eds. Cambridge, MA: MIT Press, 1991, pp. 67–95.
[34] P. J. Werbos, "Approximate dynamic programming for real-time control and neural modeling," in *Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches*, D. A. White and D. A. Sofge, Eds. New York: Van Nostrand, 1992, ch. 13.
[35] Q. Yang, J. B. Vance, and S. Jagannathan, "Control of nonaffine nonlinear discrete-time systems using reinforcement-learning-based linearly parameterized neural networks," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 38, no. 4, pp. 994–1001, Aug. 2008.
[36] H. Zhang, Q. Wei, and Y. Luo, "A novel infinite-time optimal tracking control scheme for a class of discrete-time nonlinear systems via the greedy HDP iteration algorithm," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 38, no. 4, pp. 937–942, Aug. 2008.
[37] H. Zhang, Q. Wei, and D. Liu, "An iterative adaptive dynamic programming method for solving a class of nonlinear zero-sum differential games," *Automatica*, vol. 47, no. 1, pp. 207–214, Jan. 2011.

Derong Liu (S'91–M'94–SM'96–F'05) received the B.S. degree in mechanical engineering from East China Institute of Technology (now Nanjing University of Science and Technology), Nanjing, China, in 1982; the M.S. degree in automatic control theory and applications from the Chinese Academy of Sciences, Beijing, China, in 1987; and the Ph.D. degree in electrical engineering from the University of Notre Dame, Notre Dame, IN, in 1994.

He was a Product Design Engineer with China North Industries Corporation, Jilin, China, from 1982 to 1984. He was an Instructor with the Graduate School of the Chinese Academy of Sciences from 1987 to 1990. He was a Staff Fellow with General Motors Research and Development Center, General Motors Corporation, 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, 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 10 books (five research monographs and five edited volumes).

Prof. Liu is an Elected Member of the Board of Governors of the International Neural Network Society. He was a member of the Conference Editorial Board of the IEEE Control Systems Society (1995–2000). He was an Elected AdCom Member of the IEEE Computational Intelligence Society (2006–2008). He was the General Chair of the 2007 International Symposium on Neural Networks (Nanjing, China); the 2009 IEEE Conference on Service Operations, Logistics, and Informatics (Chicago, IL); and the 2008 IEEE International Conference on Networking, Sensing and Control (Sanya, China). He is the General Chair of the 2014 IEEE World Congress on Computational Intelligence (Beijing, China). He serves as an Associate Editor of Neurocomputing and the International Journal of Neural Systems. He was an Associate Editor of Automatica (2006–2009), the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS-I: FUNDAMENTAL THEORY AND APPLICATIONS (1997–1999), the IEEE TRANSACTIONS ON SIGNAL PROCESSING (2001–2003), the IEEE TRANSACTIONS ON NEURAL NETWORKS (2004–2009), the *IEEE Computational Intelligence Magazine* (2006–2009), and the *IEEE Circuits and Systems Magazine* (2008–2009). He was the Letters Editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS (2006–2008). He was the Founding Editor of the IEEE Computational Intelligence Society's *Electronic Letter* (2004–2009). 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. He was a recipient the Michael J. Birck Fellowship from the University of Notre Dame (1990), the Harvey N. Davis Distinguished Teaching Award from Stevens Institute of Technology (1997), the Faculty Early Career Development (CAREER) Award from the National Science Foundation (1999), the University Scholar Award from the University of Illinois (2006), and the Overseas Outstanding Young Scholar Award from the National Natural Science Foundation of China (2008). He is also a member of Eta Kappa Nu.

Qinglai Wei received the B.S. degree in automation, the M.S. degree in control theory and control engineering, and the Ph.D. degree in control theory and control engineering, from Northeastern University, Shenyang, China, in 2002, 2005, and 2008, respectively.

From 2009 to 2011, he was a Postdoctoral Fellow with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, China. He is currently an Assistant Research Fellow with the same institute. His research interests include neural-networks-based control, adaptive dynamic programming, optimal control, nonlinear system, and their industrial applications.