Information Sciences 220 (2013) 331–342

Contents lists available at SciVerse ScienceDirect

Information Sciences

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

An iterative adaptive dynamic programming algorithm for optimal control of unknown discrete-time nonlinear systems with constrained inputs

Derong Liu *, Ding Wang, Xiong Yang

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

ARTICLE INFO

Article history:
Received 2 November 2011
Received in revised form 3 April 2012
Accepted 15 July 2012
Available online 31 July 2012

Keywords:
Adaptive dynamic programming
Approximate dynamic programming
Control constraints
Globalized dual heuristic programming
Neural networks
Optimal control

ABSTRACT

In this paper, the adaptive dynamic programming (ADP) approach is employed for designing an optimal controller of unknown discrete-time nonlinear systems with control constraints. A neural network is constructed for identifying the unknown dynamical system with stability proof. Then, the iterative ADP algorithm is developed to solve the optimal control problem with convergence analysis. Two other neural networks are introduced for approximating the cost function and its derivatives and the control law, under the framework of globalized dual heuristic programming technique. Furthermore, two simulation examples are included to verify the theoretical results.

1. Introduction

The nonlinear optimal control has been the focus of control fields for many decades [8,16]. It often needs to solve the nonlinear Hamilton–Jacobi–Bellman (HJB) equation. For instance, the discrete-time HJB (DTHJB) equation is more difficult to work with than the Riccati equation because it involves solving nonlinear partial difference equations. Although dynamic programming has been a useful technique in handling optimal control problems for many years, it is often computationally untenable to perform it to obtain the optimal solutions [4].

Effective techniques have been employed to construct learning systems [22,20,37,19,35,3,12,11]. Characterized by strong abilities of self-learning and adaptivity, artificial neural networks (ANN or NN) are also a functional tool to implement learning control [33,15,13,34]. Additionally, they are often used to carry out universal function approximation in adaptive/approximate dynamic programming (ADP) algorithms. The ADP method was proposed by Werbos [33,34] to deal with optimal control problems forward-in-time. There were several synonyms used for ADP, including "adaptive critic designs" [21], "adaptive dynamic programming" [30,17], "approximate dynamic programming" [34,24,2], "neuro-dynamic programming" [5], "neural dynamic programming" [23], and "reinforcement learning" [6].

In recent years, ADP and related research have gained much attention from researchers [1,2,5,6,9,10,14,17,18,21,23–32,34,36]. According to [21] and [34], ADP approaches were classified into several main schemes: heuristic dynamic programming (HDP), action-dependent HDP (ADHDP), also known as Q-learning, dual heuristic dynamic programming (DHP), ADDHP, globalized DHP (GDHP), and ADGDHP. Al-Tamimi et al. [2] proposed a greedy HDP iteration algorithm to

* Corresponding author. Tel.: +86 10 62557379; fax: +86 10 62650912.
E-mail addresses: derong.liu@ia.ac.cn (D. Liu), ding.wang@ia.ac.cn (D. Wang), xiong.yang@ia.ac.cn (X. Yang).

0020-0255/$ - see front matter © 2012 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.ins.2012.07.006

solve the DTHJB equation of optimal control of discrete-time affine nonlinear systems. Abu-Khalaf and Lewis [1], Vrabie and Lewis [27], and Vamvoudakis and Lewis [25] investigated the continuous-time nonlinear optimal control problems based on the idea of ADP.

With the increasing complexity of industry processes, the data-based method has achieved great interest among control engineers. It does not need to build accurate mathematical models of controlled plants and thus has significant practical value. Kim and Lewis [14] presented a model-free H_∞ control design scheme for unknown linear discrete-time systems via Q-learning, which was expressed in the form of linear matrix inequality. Campi and Savaresi [7] proposed a virtual reference feedback tuning approach which was in fact a data-based method. In this paper, we solve the constrained optimal control problem of unknown discrete-time nonlinear systems based on the iterative ADP algorithm via GDHP technique (i.e., iterative GDHP algorithm). An NN model is constructed as an identifier to learn the unknown controlled plant. Then, the iterative ADP algorithm is introduced to solve the DTHJB equation with convergence proof. Next, the optimal controller can be designed by employing the GDHP technique.

This paper is organized as follows: In Section 2, the optimal control problem and the DTHJB equation are recalled for discrete-time nonlinear systems. In Section 3, we first design an NN identifier for unknown controlled system with stability proof. Then, the optimal control scheme based on the iterative ADP algorithm is developed with convergence analysis. In Section 4, the implementation of iterative ADP algorithm is presented through NN-based GDHP technique. In Section 5, two numerical examples are given to demonstrate the effectiveness of the proposed optimal control scheme. In Section 6, concluding remarks are given.

2. Preliminaries

In this paper, we study the nonlinear discrete-time systems described by

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

where $x_k \in \mathbb{R}^n$ is the state vector and $u_k = u(x_k) \in \mathbb{R}^m$ is the control vector. Let the initial state be denoted by x_0. The system function $F(x_k, u_k)$ is continuous for $\forall x_k, u_k$ and $F(0,0) = 0$. Hence, $x = 0$ is an equilibrium state of system (1) under control $u = 0$. We define $\Omega_u = \{u_k | u_k = [u_{1k}, u_{2k}, \ldots u_{mk}]^T \in \mathbb{R}^m, |u_{ik}| \leqslant \bar{u}_i, i = 1, 2, \ldots m\}$, where $\bar{u}_i$ is the saturating bound for the ith actuator. Let $\overline{U} = \mathrm{diag}\{\bar{u}_1, \bar{u}_2, \ldots \bar{u}_m\}$ be a constant diagonal matrix.

The objective for general optimal control problems is to find the control law $u(x)$ which minimizes the infinite horizon cost function given by

$$J(x_k) = \sum_{i=k}^{\infty} U(x_i, u_i),$$

where U is the utility function, $U(0, 0) = 0$, and $U(x_i, u_i) \geqslant 0$ for $\forall x_i, u_i$. According to Bellman's optimality principle, the optimal cost function

$$J^*(x_k) = \min_{u_k, u_{k+1}, \ldots u_\infty} \sum_{i=k}^{\infty} U(x_i, u_i)$$

can be rewritten as

$$J^*(x_k) = \min_{u_k} \left\{ U(x_k, u_k) + \min_{u_{k+1}, \ldots u_\infty} \sum_{i=k+1}^{\infty} U(x_i, u_i) \right\}.$$

In other words, $J^*(x_k)$ satisfies the DTHJB equation

$$J^*(x_k) = \min_{u_k} \{U(x_k, u_k) + J^*(x_{k+1})\}. \tag{2}$$

The corresponding optimal control u^* is

$$u^*(x_k) = \arg\min_{u_k} \{U(x_k, u_k) + J^*(x_{k+1})\}. \tag{3}$$

In many literatures [2,9,28], the utility function is chosen as

$$U(x_i, u_i) = x_i^T Q x_i + u_i^T R u_i, \tag{4}$$

where Q and R are positive definite matrices with suitable dimensions. However, when dealing with constrained optimal control problems, it is not the case any more. Inspired by the work of [1,36], we can employ a generalized non-quadratic functional

$$Y(u_i) = 2 \int_0^{u_i} \psi^{-T}(\overline{U}^{-1} s) \overline{U} R \mathrm{d}s \tag{5}$$

to substitute the quadratic term of u_i in (4). Note that in (5), $\psi^{-1}(u_i) = [\phi^{-1}(u_{1i}), \phi^{-1}(u_{2i}), \ldots, \phi^{-1}(u_{mi})]^T$, R is positive definite and assumed to be diagonal for simplicity of analysis, $s \in \mathbb{R}^m, \psi \in \mathbb{R}^m, \psi^{-T}$ denotes $(\psi^{-1})^T$, and $\phi(\cdot)$ is a bounded one-to-one function satisfying $|\phi(\cdot)| \leqslant 1$ and belonging to $C^p(p \geqslant 1)$ and $L_2(\Omega)$. Moreover, it is a monotonic odd function with its first derivative bounded by a constant M. The well-known hyperbolic tangent function $\phi(\cdot) = \tanh(\cdot)$ is one example of such function. Besides, it is important to note that $Y(u_i)$ is positive definite since $\phi^{-1}(\cdot)$ is a monotonic odd function and R is positive definite.

In this sense, the utility function becomes $U(x_i, u_i) = x_i^T Q x_i + Y(u_i)$. Accordingly, (2) and (3) becomes

$$J^*(x_k) = \min_{u_k}\left\{x_k^T Q x_k + 2\int_0^{u_k} \psi^{-T}(\overline{U}^{-1}s)\overline{U}R\mathrm{d}s + J^*(x_{k+1})\right\}$$

and

$$u^*(x_k) = \arg\min_{u_k}\left\{x_k^T Q x_k + 2\int_0^{u_k} \psi^{-T}(\overline{U}^{-1}s)\overline{U}R\mathrm{d}s + J^*(x_{k+1})\right\},$$

respectively.

3. Neural optimal control scheme based on the iterative ADP algorithm

In this section, we present the neural optimal control scheme for unknown controlled system using the iterative ADP algorithm. Three subsections are embodied, including the NN identification of the unknown controlled plant, the derivation of the iterative ADP algorithm, and the convergence proof of the iterative algorithm.

3.1. Identification of the unknown controlled system using NN

In this section, a three-layer feedforward NN is constructed to identify the unknown system dynamics. Let the number of hidden layer neurons be denoted by l, the ideal weight matrix between the input layer and hidden layer be denoted by v_m^*, and the ideal weight matrix between the hidden layer and output layer be denoted by ω_m^*. According to the universal approximation property [13] of NN, the system dynamics (1) has an NN representation on a compact set S, which can be written as

$$x_{k+1} = \omega_m^{*T}\sigma\left(v_m^{*T} z_k\right) + \varepsilon_k. \tag{6}$$

In (6), $z_k = [x_k^T u_k^T]^T$ is the NN input, ε_k is the bounded NN functional approximation error, and $[\sigma(\xi)]_i = (e^{\xi_i} - e^{-\xi_i})/(e^{\xi_i} + e^{-\xi_i}), i = 1, 2, \ldots l$ are the activation functions. Let $\bar{z}_k = v_m^{*T} z_k, \bar{z}_k \in \mathbb{R}^l$. The selected activation functions are bounded such that $\|\sigma(\bar{z}_k)\| \leqslant \sigma_M$ for a constant σ_M.

In this paper, we define the NN system identification scheme as

$$\hat{x}_{k+1} = \omega_m^T(k)\sigma(\bar{z}_k) - r_k, \tag{7}$$

where $\hat{x}_k$ is the estimated system state vector, r_k is the robust term, and $\omega_m(k)$ is the estimation of the constant ideal weight matrix.

Denote $\tilde{x}_k = \hat{x}_k - x_k$ as the system identification error. Then, combining (6) with (7), we obtain the identification error dynamics

$$\tilde{x}_{k+1} = \tilde{\omega}_m^T(k)\sigma(\bar{z}_k) - r_k - \varepsilon_k, \tag{8}$$

where $\tilde{\omega}_m(k) = \omega_m(k) - \omega_m^*$. Inspired by the work of [9], we define the robust term as a function of the identification error $\tilde{x}_k$ and an additional tunable parameter $\beta(k) \in \mathbb{R}$, i.e., $r_k = \beta(k)\tilde{x}_k/(\tilde{x}_k^T\tilde{x}_k + C)$, where $C > 0$ is a constant. Denote β^* as the constant ideal value of the parameter $\beta(k)$. Besides, let $\tilde{\beta}(k) = \beta(k) - \beta^*, \psi_k = \tilde{\omega}_m^T(k)\sigma(\bar{z}_k)$, and $\varphi_k = \tilde{\beta}(k)\tilde{x}_k/(\tilde{x}_k^T\tilde{x}_k + C)$. Then, the system dynamics (8) can be rewritten as

$$\tilde{x}_{k+1} = \psi_k - \varphi_k - \frac{\beta^*\tilde{x}_k}{\tilde{x}_k^T\tilde{x}_k + C} - \varepsilon_k. \tag{9}$$

The parameters in the system identification process are updated to minimize the following performance measure: $E_{k+1} = 0.5\tilde{x}_{k+1}^T\tilde{x}_{k+1}$. Using the gradient-based adaptation rule, the NN weight and tunable parameter can be updated by

$$\omega_m(k+1) = \omega_m(k) - \alpha_m\left[\frac{\partial E_{k+1}}{\partial\omega_m(k)}\right] = \omega_m(k) - \alpha_m\sigma(\bar{z}_k)\tilde{x}_{k+1}^T, \tag{10}$$

$$\beta(k+1) = \beta(k) - \alpha_r\left[\frac{\partial E_{k+1}}{\partial\beta(k)}\right] = \beta(k) + \alpha_r\frac{\tilde{x}_{k+1}^T\tilde{x}_k}{\tilde{x}_k^T\tilde{x}_k + C}, \tag{11}$$

where $\alpha_m > 0$ and $\alpha_r > 0$ are the learning rates.

Before presenting the stability proof of the error dynamics, we give the following assumption, which has been used in [13,9].

Assumption 1. The NN approximation error term ε_k is assumed to be upper bounded by a function of the state estimation error $\tilde{x}_k$, i.e.,

$$\varepsilon_k^{\mathrm{T}}\varepsilon_k \leqslant \varepsilon_{Mk} = \delta\tilde{x}_k^{\mathrm{T}}\tilde{x}_k, \tag{12}$$

where δ is a bounded constant value such that $\|\delta\| \leqslant \delta_M$.

Theorem 1. *Let the identification scheme (7) be used to identify the nonlinear system (1), and let the parameter update law given in (10) and (11) be used for tuning the NN weights and the robust term, respectively. Then, the state estimation error $\tilde{x}_k$ is asymptotically stable while the parameter estimation error $\tilde{\omega}_m(k)$ and $\tilde{\beta}(k)$ are bounded.*

Proof. Consider the positive definite Lyapunov function candidate defined as

$$L_k = L_{1k} + L_{2k} + L_{3k}, \tag{13}$$

where

$$L_{1k} = \tilde{x}_k^{\mathrm{T}}\tilde{x}_k,\ L_{2k} = \frac{\tilde{\beta}^2(k)}{\alpha_r},\ L_{3k} = \frac{1}{\alpha_m}\mathrm{tr}\{\tilde{\omega}_m^{\mathrm{T}}(k)\tilde{\omega}_m(k)\}.$$

In the following, we denote $C_k = \tilde{x}_k^{\mathrm{T}}\tilde{x}_k + C$ for brief. By taking the first difference of the Lyapunov function (13) and substituting the identification error dynamics (9) and the parameter update law (10) and (11), we can derive that

$$\begin{aligned}
\Delta L_{1k} &= \tilde{x}_{k+1}^{\mathrm{T}}\tilde{x}_{k+1} - \tilde{x}_k^{\mathrm{T}}\tilde{x}_k = \psi_k^{\mathrm{T}}\psi_k + \varphi_k^{\mathrm{T}}\varphi_k + \varepsilon_k^{\mathrm{T}}\varepsilon_k - \tilde{x}_k^{\mathrm{T}}\tilde{x}_k - 2\psi_k^{\mathrm{T}}\varphi_k - 2\psi_k^{\mathrm{T}}\varepsilon_k + 2\varphi_k^{\mathrm{T}}\varepsilon_k \\
&\quad - \frac{2\beta^*\psi_k^{\mathrm{T}}\tilde{x}_k}{C_k} + \frac{2\beta^*\varphi_k^{\mathrm{T}}\tilde{x}_k}{C_k} + \frac{2\beta^*\varepsilon_k^{\mathrm{T}}\tilde{x}_k}{C_k} + \frac{\beta^{*2}\tilde{x}_k^{\mathrm{T}}\tilde{x}_k}{C_k^2}, \\
\Delta L_{2k} &= \frac{\tilde{\beta}^2(k+1) - \tilde{\beta}^2(k)}{\alpha_r} = 2\left(\psi_k^{\mathrm{T}} - \varphi_k^{\mathrm{T}} - \frac{\beta^*\tilde{x}_k^{\mathrm{T}}}{C_k} - \varepsilon_k^{\mathrm{T}}\right)\varphi_k + \alpha_r\left(\frac{\tilde{x}_{k+1}^{\mathrm{T}}\tilde{x}_k}{C_k}\right)^2.
\end{aligned}$$

For ΔL_{3k}, we apply the Cauchy–Schwarz inequality, and then obtain

$$\Delta L_{3k} = \frac{1}{\alpha_m}\mathrm{tr}\{\tilde{\omega}_m^{\mathrm{T}}(k+1)\tilde{\omega}_m(k+1) - \tilde{\omega}_m^{\mathrm{T}}(k)\tilde{\omega}_m(k)\} \leqslant -2\psi_k^{\mathrm{T}}\tilde{x}_{k+1} + 4\alpha_m\sigma^{\mathrm{T}}(\bar{z}_k)\sigma(\bar{z}_k)\left(\psi_k^{\mathrm{T}}\psi_k + \varphi_k^{\mathrm{T}}\varphi_k + \varepsilon_k^{\mathrm{T}}\varepsilon_k + \frac{\beta^{*2}\tilde{x}_k^{\mathrm{T}}\tilde{x}_k}{C_k^2}\right).$$

Noting that $\Delta L_k = \Delta_{1k} + \Delta L_{2k} + \Delta_{3k}$ and considering $\|\sigma(\bar{z}_k)\| \leqslant \sigma_M$ and (12), we can find that

$$\begin{aligned}
\Delta L_k &\leqslant -\left(1 - 4\alpha_m\sigma_M^2 - 4\alpha_r\right)\left(\|\psi_k\|^2 + \|\varphi_k\|^2\right) - \left(1 - 2\delta_M - 2\delta_M^2 - 4\alpha_m\delta_M\sigma_M^2 - 4\alpha_m\delta_M^2\sigma_M^2 - 4\alpha_r\delta_M - 4\alpha_r\delta_M^2\right)\|\tilde{x}_k\|^2 \\
&\quad + 2\|\psi_k\|\|\varphi_k\|.
\end{aligned} \tag{14}$$

Then, we define $\theta_1\bar{\psi}_k = \psi_k, \theta_2\bar{\varphi}_k = \varphi_k$, where θ_1 and θ_2 are constants. After selecting the parameters as $\alpha_m\sigma_M^2 = \alpha_r, \theta_1\theta_2 = \rho, 8\alpha_m\sigma_M^2 \leqslant \theta_1^2$, and applying the Cauchy–Schwarz inequality, (14) becomes

$$\begin{aligned}
\Delta L_k &\leqslant -\left(1 - \theta_1^2 - \frac{\rho}{\theta_1^2}\right)\|\psi_k\|^2 - \left(1 - \theta_1^2 - \frac{\theta_1^2}{\rho}\right)\|\varphi_k\|^2 - \left(1 - 2\delta_M - 2\delta_M^2 - \delta_M\theta_1^2 - \delta_M^2\theta_1^2\right)\|\tilde{x}_k\|^2 \\
&= -\left(1 - \theta_1^2 - \frac{\rho}{\theta_1^2}\right)\left\|\tilde{\omega}_m^{\mathrm{T}}(k)\sigma(\bar{z}_k)\right\|^2 - \left(1 - \theta_1^2 - \frac{\theta_1^2}{\rho}\right)\|\tilde{\beta}(k)\|^2\left\|\frac{\tilde{x}_k}{C_k}\right\|^2 - \left(1 - 2\delta_M - 2\delta_M^2 - \delta_M\theta_1^2 - \delta_M^2\theta_1^2\right)\|\tilde{x}_k\|^2.
\end{aligned} \tag{15}$$

From (15), we can conclude that $\Delta L_k \leqslant 0$ if $0 < \delta_M \leqslant \left(\sqrt{3} - 1\right)/2, 0 < \rho < 1/4$, and $\tau_1 < \theta_1 < \min\{\tau_2, \tau_3, \tau_4\}$, where

$$\tau_1 = \sqrt{\frac{1 - \sqrt{1 - 4\rho}}{2}},\ \tau_2 = \sqrt{\frac{1 - 2\delta_M - 2\delta_M^2}{\delta_M + \delta_M^2}},\ \tau_3 = \sqrt{\frac{\rho}{1 + \rho}},\ \tau_4 = \sqrt{\frac{1 + \sqrt{1 - 4\rho}}{2}}.$$

As long as the parameters are selected as above, $\Delta L_k \leqslant 0$ in (15), which shows the stability of error dynamics in the sense of Lyapunov. Therefore, $\tilde{x}_k$, $\tilde{\omega}_m(k)$, and $\tilde{\beta}(k)$ are bounded, provided $\tilde{x}_0, \tilde{\omega}_m(0)$, and $\tilde{\beta}(0)$ are bounded in the compact set S. Furthermore, by summing both sides of (15) to infinity and taking the absolute value, we can obtain

$$\begin{aligned}
&\sum_{k=0}^{\infty}\left\{\left(1 - \theta_1^2 - \frac{\rho}{\theta_1^2}\right)\left\|\tilde{\omega}_m^{\mathrm{T}}(k)\sigma(\bar{z}_k)\right\|^2 + \left(1 - \theta_1^2 - \frac{\theta_1^2}{\rho}\right)\|\tilde{\beta}(k)\|^2\left\|\frac{\tilde{x}_k}{C_k}\right\|^2 + \left(1 - 2\delta_M - 2\delta_M^2 - \delta_M\theta_1^2 - \delta_M^2\theta_1^2\right)\|\tilde{x}_k\|^2\right\} \\
&\quad \leqslant \left|\sum_{k=0}^{\infty}\Delta L_k\right| = |\lim_{k\to\infty} L_k - L_0| < \infty.
\end{aligned} \tag{16}$$

From (16), we can conclude that $\|\tilde{x}_k\| \to 0$ as $k \to \infty$. □

According to Theorem 1, after a sufficient learning process, the NN system identification error converges to zero and the robust term approaches zero as well, i.e., we have

$$x_{k+1} = \omega_m^{\mathrm{T}}(k)\sigma(\bar{z}_k). \tag{17}$$

Then, we can further derive that

$$\frac{\partial x_{k+1}}{\partial u_k} = \frac{\partial\left(\omega_m^{\mathrm{T}}(k)\sigma(\bar{z}_k)\right)}{\partial u_k} = \omega_m^{\mathrm{T}}(k)\sigma'(\bar{z}_k)v_m^{*\mathrm{T}}\Theta, \tag{18}$$

where

$$\sigma'(\bar{z}_k) = \frac{\partial\sigma(\bar{z}_k)}{\partial\bar{z}_k}, \Theta = \frac{\partial z_k}{\partial u_k} = \frac{0_{n\times m}}{I_m}$$

and I_m is an $m \times m$ identity matrix.

Since it is difficult to get $\partial x_{k+1}/\partial u_k$ otherwise, we can use (18) instead when solving the optimal control via (3). Next, this result will be used in the derivation and implementation of the iterative ADP algorithm.

3.2. Derivation of the iterative algorithm

The iterative ADP algorithm is performed as follows. First, we start with the initial cost function $V_0(\cdot) = 0$ and solve

$$v_0(x_k) = \operatorname{argmin}_{u_k}\{U(x_k,u_k) + V_0(x_{k+1})\} = \operatorname{argmin}_{u_k}\{U(x_k,u_k) + V_0(F(x_k,u_k))\}. \tag{19}$$

Then, we update the cost function by

$$V_1(x_k) = \min_{u_k}\{U(x_k,u_k) + V_0(x_{k+1})\} = U(x_k, v_0(x_k)) + V_0(F(x_k, v_0(x_k))). \tag{20}$$

Next, for $i = 1, 2, \ldots$, the algorithm iterates between

$$v_i(x_k) = \operatorname{argmin}_{u_k}\{U(x_k,u_k) + V_i(x_{k+1})\} = \operatorname{argmin}_{u_k}\{U(x_k,u_k) + V_i(F(x_k,u_k))\} \tag{21}$$

and

$$V_{i+1}(x_k) = \min_{u_k}\{U(x_k,u_k) + V_i(x_{k+1})\} = U(x_k, v_i(x_k)) + V_i(F(x_k, v_i(x_k))). \tag{22}$$

In the following, we will present the convergence proof of the iteration (19)–(22) with the cost function $V_i \to J^*$ and the control law $v_i \to u^*$ as $i \to \infty$.

3.3. Convergence analysis of the iterative algorithm

Lemma 1. *Let $\{v_i\}$ be the control laws defined as in (21) and $\{\mu_i\}$ be any arbitrary sequence of control laws. Define V_i as in (22) and Λ_i as*

$$\Lambda_{i+1}(x_k) = U(x_k, \mu_i(x_k)) + \Lambda_i(x_{k+1}). \tag{23}$$

If $V_0(\cdot) = \Lambda_0(\cdot) = 0$, then $V_{i+1}(x) \leqslant \Lambda_{i+1}(x)$, $\forall i$.

Proof. Since $V_0(\cdot) = \Lambda_0(\cdot) = 0$, we have

$$V_1(x_k) = \min_{u_k}\{U(x_k,u_k)\} \leqslant U(x_k, \mu_0(x_k)) = \Lambda_1(x_k). \tag{24}$$

It reveals that $V_1(x) \leqslant \Lambda_1(x)$ since (24) is true for any x_k. Next, we assume that $V_i(x) \leqslant \Lambda_i(x)$. Then, we have $V_i(x_{k+1}) \leqslant \Lambda_i(x_{k+1})$. According to (22) and (23), we can obtain that

$$V_{i+1}(x_k) \leqslant \min_{u_k}\{U(x_k,u_k) + \Lambda_i(x_{k+1})\} \leqslant \Lambda_{i+1}(x_k). \tag{25}$$

It implies $V_{i+1}(x) \leqslant \Lambda_{i+1}(x)$ because (25) is true for any x_k. Thus, we complete the proof by mathematical induction. □

Lemma 2. *Let the sequence $\{V_i\}$ be defined as in (22). If the system is controllable, then there is an upper bound Y such that $0 \leqslant V_i(x_k) \leqslant Y$, $\forall i$.*

Proof. Let $\eta(x_k)$ be any admissible control input. Let V_i be updated as in (22) and Z_i be updated by $Z_{i+1}(x_k) = U(x_k, \eta(x_k)) + Z_i(x_{k+1})$, where $V_0(\cdot) = Z_0(\cdot) = 0$. Clearly, $Z_1(x_k) = U(x_k, \eta(x_k))$. By observing

$$\begin{aligned} Z_{i+1}(x_k) - Z_i(x_k) &= Z_i(x_{k+1}) - Z_{i-1}(x_{k+1}) = Z_{i-1}(x_{k+2}) - Z_{i-2}(x_{k+2}) = Z_{i-2}(x_{k+3}) - Z_{i-3}(x_{k+3}) \ldots = Z_1(x_{k+i}) - Z_0(x_{k+i}) \\ &= Z_1(x_{k+i}), \end{aligned}$$

we have

$$Z_{i+1}(x_k) = Z_1(x_{k+i}) + Z_i(x_k) = Z_1(x_{k+i}) + Z_1(x_{k+i-1}) + Z_{i-1}(x_k) = Z_1(x_{k+i}) + Z_1(x_{k+i-1}) + Z_1(x_{k+i-2}) + Z_{i-2}(x_k)$$
$$= Z_1(x_{k+i}) + Z_1(x_{k+i-1}) + Z_1(x_{k+i-2}) + \cdots + Z_1(x_{k+1}) + Z_1(x_k).$$

Therefore, we obtain

$$Z_{i+1}(x_k) = \sum_{j=0}^{i} Z_1(x_{k+j}) = \sum_{j=0}^{i} U(x_{k+j}, \eta(x_{k+j})).$$

Since $\eta(x_k)$ is an admissible control input, we have

$$Z_{i+1}(x_k) \leqslant \sum_{j=0}^{\infty} U(x_{k+j}, \eta(x_{k+j})) \leqslant Y, \forall i.$$

By using Lemma 1, we can obtain $V_{i+1}(x_k) \leqslant Z_{i+1}(x_k) \leqslant Y$, $\forall i$, and thus complete the proof. □

Based on Lemmas 1 and 2, we now present our main results.

Theorem 2. *Define the sequence $\{V_i\}$ as in (22) with $V_0(\cdot) = 0$, and the control law sequence $\{v_i\}$ as in (21). Then, $\{V_i\}$ is a nondecreasing sequence satisfying $V_i \leqslant V_{i+1}, \forall i$.*

Proof. Define a new sequence $\Phi_{i+1}(x_k) = U(x_k, v_{i+1}(x_k)) + \Phi_i(x_{k+1})$ with $\Phi_0(\cdot) = V_0(\cdot) = 0$. Next, we prove that $\Phi_i(x_k) \leqslant V_{i+1}(x_k)$ by mathematical induction.

First, we prove that it holds for $i = 0$. Since $V_1(x_k) - \Phi_0(x_k) = U(x_k, v_0(x_k)) \geqslant 0$, we have $\Phi_0(x_k) \leqslant V_1(x_k)$. Second, we assume that it holds for $i-1$, i.e., $\Phi_{i-1}(x_k) \leqslant V_i(x_k)$, $\forall x_k$. Then, for i, by noticing $V_{i+1}(x_k) = U(x_k, v_i(x_k)) + V_i(x_{k+1})$ and $\Phi_i(x_k) = U(x_k, v_i(x_k)) + \Phi_{i-1}(x_{k+1})$, we can get $V_{i+1}(x_k) - \Phi_i(x_k) = V_i(x_{k+1}) - \Phi_{i-1}(x_{k+1}) \geqslant 0$, i.e., $\Phi_i(x_k) \leqslant V_{i+1}(x_k)$. Thus, we complete the proof by mathematical induction. Furthermore, from Lemma 1, we know that $V_i(x_k) \leqslant \Phi_i(x_k)$. Therefore, we have

$$V_i(x_k) \leqslant \Phi_i(x_k) \leqslant V_{i+1}(x_k) \tag{26}$$

and also complete the proof. □

In light of Lemma 2 and Theorem 2, the limit of the cost function sequence $\{V_i\}$ exists when $i \to \infty$. The same is true for the sequence $\{v_i\}$ according to (3) and (21). Here, we denote $V_\infty(x_k) = \lim_{i\to\infty} V_i(x_k)$ and $v_\infty(x_k) = \lim_{i\to\infty} v_i(x_k)$, respectively. Next, we give the following theorem.

Theorem 3. *Let the cost function sequence $\{V_i\}$ be defined as in (22) and $V_\infty(x_k)$ be its limit. Then, we have*

$$V_\infty(x_k) = \min_{u_k}\{U(x_k, u_k) + V_\infty(x_{k+1})\}. \tag{27}$$

Proof. On one hand, for any u_k and i, according to (22), we can derive

$$V_i(x_k) \leqslant U(x_k, u_k) + V_{i-1}(x_{k+1}). \tag{28}$$

Combining (28) with

$$V_i(x_k) \leqslant V_\infty(x_k), \forall i, \tag{29}$$

which is obtained from (26), we have $V_i(x_k) \leqslant U(x_k, u_k) + V_\infty(x_{k+1}), \forall i$. By letting $i \to \infty$, we obtain

$$V_\infty(x_k) \leqslant U(x_k, u_k) + V_\infty(x_{k+1}). \tag{30}$$

Since u_k is chosen arbitrarily in (30), we have

$$V_\infty(x_k) \leqslant \min_{u_k}\{U(x_k, u_k) + V_\infty(x_{k+1})\}. \tag{31}$$

On the other hand, since the cost function sequence satisfies $V_i(x_k) = \min_{u_k}\{U(x_k, u_k) + V_{i-1}(x_{k+1})\}$ for any i, considering (29), we have $V_\infty(x_k) \geqslant \min_{u_k}\{U(x_k, u_k) + V_{i-1}(x_{k+1})\}, \forall i$. By letting $i \to \infty$, we can get

$$V_\infty(x_k) \geqslant \min_{u_k}\{U(x_k, u_k) + V_\infty(x_{k+1})\}. \tag{32}$$

Based on (31) and (32), we can conclude that (27) is true. □

By observing (2) and (27), we obtain $V_\infty = J^*$, which implies that the cost function sequence converges to the optimal cost function of the DTHJB equation. Additionally, the control law related to V_∞ can be formulated by

$$v_\infty(x_k) = \arg\min_{u_k}\{U(x_k, u_k) + V_\infty(x_{k+1})\}. \tag{33}$$

By making a comparison between (3) and (33), we can further derive that $v_\infty = u^*$. Therefore, we acquire the main conclusion $V_i \to J^*$ and $v_i \to u^*$ as $i \to \infty$.

4. Implementation of the iterative ADP algorithm via GDHP technique

In this section, we implement the iterative ADP algorithm described by (19)–(22) via GDHP technique. The idea is to take the function approximation structure, such as NN, to approximate both $V_i(x_k)$ and $v_i(x_k)$.

4.1. The iterative GDHP algorithm

In the iterative GDHP algorithm, there are three NNs, which are model network, critic network, and action network. In this paper, all the NNs are chosen as three-layer feedforward ones. It is important to note that the critic network of GDHP technique outputs both the cost function $J(x_k)$ and its derivative $\partial J(x_k)/\partial x_k$ [21], which is schematically depicted in Fig. 1. Similar to the critic network, the input of action network is also x_k. However, the input of model network consists of x_k and $\hat{v}_{i-1}(x_k)$. The whole structural diagram of the iterative GDHP algorithm is shown in Fig. 2, where

$$\mathrm{DER} = \left(\frac{\partial \hat{x}_{k+1}}{\partial x_k} + \frac{\partial \hat{x}_{k+1}}{\partial \hat{v}_{i-1}(x_k)}\frac{\partial \hat{v}_{i-1}(x_k)}{\partial x_k}\right)^{\mathrm{T}}.$$

According to Fig. 2, we observe that the outputs of critic network of the iterative GDHP algorithm contain not only the cost function but also its derivatives. This is a significant property of the iterative GDHP algorithm since the information associated with the cost function is as useful as the knowledge of its derivatives. Therefore, when training the critic network of the iterative GDHP algorithm, we should utilize an error measure which is a combination of the error measures of HDP and DHP. Consequently, the resulting behavior is expected to be superior to simple ADP methods.

4.2. The training process

The training of model network is completed after the system identification process and its weights are kept unchanged. Then, according to Theorem 1, when given x_k and $\hat{v}_{i-1}(x_k)$, we can compute $\hat{x}_{k+1}$ by (7). As a result, we avoid the requirement of knowing $F(x_k, u_k)$ during the implementation of the iterative GDHP algorithm.

Next, the learned NN model will be used in the training process of critic network and action network.

We define $\lambda_i(x_k) = \partial V_i(x_k)/\partial x_k$ during the training process. Hence, the critic network is used to approximate both $V_i(x_k)$ and $\lambda_i(x_k)$. The output of critic network is expressed as

$$\begin{bmatrix}\widehat{V}_i(x_k)\\ \hat{\lambda}_i(x_k)\end{bmatrix} = \begin{bmatrix}\omega_{ci}^{1\mathrm{T}}\\ \omega_{ci}^{2\mathrm{T}}\end{bmatrix}\sigma\left(v_{ci}^{\mathrm{T}}x_k\right) = \omega_{ci}^{\mathrm{T}}\sigma\left(v_{ci}^{\mathrm{T}}x_k\right),$$

where $\omega_{ci} = [\omega_{ci}^1\ \omega_{ci}^2]$. Accordingly, we have $\widehat{V}_i(x_k) = \omega_{ci}^{1\mathrm{T}}\sigma(v_{ci}^{\mathrm{T}}x_k)$ and $\hat{\lambda}_i(x_k) = \omega_{ci}^{2\mathrm{T}}\sigma(v_{ci}^{\mathrm{T}}x_k)$. The target function can be written as

$$V_i(x_k) = U(x_k, \hat{v}_{i-1}(x_k)) + \widehat{V}_{i-1}(\hat{x}_{k+1})$$

and

$$\lambda_i(x_k) = \frac{\partial U(x_k, \hat{v}_{i-1}(x_k))}{\partial x_k} + \frac{\partial \widehat{V}_{i-1}(\hat{x}_{k+1})}{\partial x_k} = 2Qx_k + 2\left(\frac{\partial \hat{v}_{i-1}(x_k)}{\partial x_k}\right)^{\mathrm{T}} R\hat{v}_{i-1}(x_k) + \left(\frac{\partial \hat{x}_{k+1}}{\partial x_k} + \frac{\partial \hat{x}_{k+1}}{\partial \hat{v}_{i-1}(x_k)}\frac{\partial \hat{v}_{i-1}(x_k)}{\partial x_k}\right)^{\mathrm{T}}\hat{\lambda}_{i-1}(\hat{x}_{k+1}).$$

Then, we define the error function of critic network as $e_{cik}^1 = \widehat{V}_i(x_k) - V_i(x_k)$ and $e_{cik}^2 = \hat{\lambda}_i(x_k) - \lambda_i(x_k)$. The objective function to be minimized in the critic network is $E_{cik} = (1-\eta)E_{cik}^1 + \eta E_{cik}^2$, where $E_{cik}^1 = 0.5e_{cik}^{1\mathrm{T}}e_{cik}^1$ and $E_{cik}^2 = 0.5e_{cik}^{2\mathrm{T}}e_{cik}^2$. The weight update rule for training critic network is also gradient-based adaptation which is given by

$$\omega_{ci}(j+1) = \omega_{ci}(j) - \alpha_c\left[(1-\eta)\frac{\partial E_{cik}^1}{\partial \omega_{ci}(j)} + \eta\frac{\partial E_{cik}^2}{\partial \omega_{ci}(j)}\right],$$

$$v_{ci}(j+1) = v_{ci}(j) - \alpha_c\left[(1-\eta)\frac{\partial E_{cik}^1}{\partial v_{ci}(j)} + \eta\frac{\partial E_{cik}^2}{\partial v_{ci}(j)}\right],$$

where $\alpha_c > 0$ is the learning rate of critic network, j is the inner-loop iteration step for updating weight parameters, and $0 \leqslant \eta \leqslant 1$ is a parameter that adjusts how HDP and DHP are combined in GDHP.

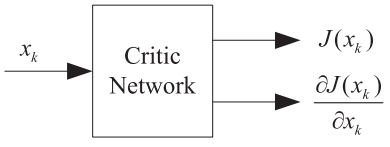

Fig. 1. The critic network of GDHP technique.

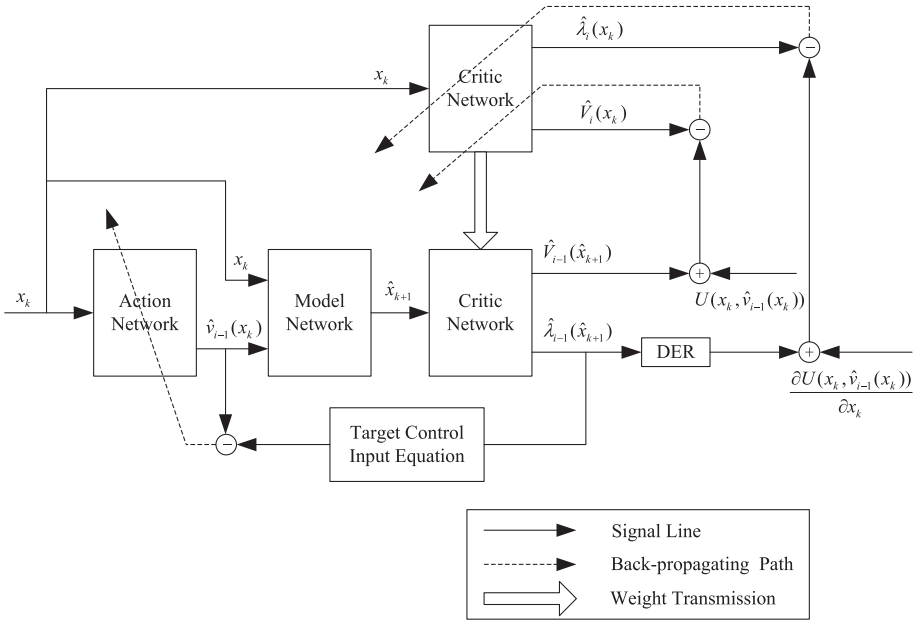

Fig. 2. The structural diagram of the iterative GDHP algorithm.

In the action network, the state x_k is used as input to obtain the approximated optimal control as output of the network, which is formulated as $\hat{v}_{i-1}(x_k) = \omega^{\mathrm{T}}_{a(i-1)}\sigma\left(v^{\mathrm{T}}_{a(i-1)}x_k\right)$. The target control input is given by

$$v_{i-1}(x_k) = \mathrm{argmin}_{u_k}\{U(x_k, u_k) + \widehat{V}_{i-1}(\hat{x}_{k+1})\}.$$

The error function of action network can be defined as $e_{a(i-1)k} = \hat{v}_{i-1}(x_k) - v_{i-1}(x_k)$. The weights of action network are updated to minimize $E_{a(i-1)k} = 0.5e^{\mathrm{T}}_{a(i-1)k}e_{a(i-1)k}$. Similarly, the weight update algorithm is

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

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

Remark 1. According to Lemma 2 and Theorems 2 and 3, $V_i \to J^*$ as $i \to \infty$. Since $\lambda_i(x_k) = \partial V_i(x_k)/\partial x_k$, we can conclude that the sequence $\{\lambda_i\}$ is also convergent with $\lambda_i \to \lambda^*$ as $i \to \infty$.

Note that some parameters, like the number of neurons, are difficult to determine in terms of theory. Thus, at the present stage, these parameters in the algorithm are mainly chosen according to experience. Meanwhile, one of our main efforts is to investigate how the parameters affect the control performances and when the best results can be derived.

5. Numerical examples

In this section, two numerical examples are provided to demonstrate the effectiveness of the control scheme derived by the iterative GDHP algorithm.

Example 1. This example is chosen from [31] with some modifications. Considering the following nonlinear system:

$$x_{k+1} = 1.2x_k + \sin(0.1x_k^2 + u_k), \tag{34}$$

where $x_k \in \mathbb{R}, u_k \in \mathbb{R}, k = 1, 2, \ldots$. Clearly, $x_k = 0$ is an equilibrium state of system (34). However, the system is unstable at this equilibrium, since $(\partial x_{k+1}/\partial x_k)|_{(0,0)} = 1.2 > 1$. It is desired to control the system with control constraint of $|u| \leqslant 1$. The cost function is chosen as

$$J(x_k) = \sum_{i=k}^{\infty}\left\{x_i^{\mathrm{T}}Qx_i + 2\int_0^{u_i}\tanh^{-\mathrm{T}}(\overline{U}^{-1}s)\overline{U}R\mathrm{d}s\right\}, \tag{35}$$

where Q and R are identity matrices with suitable dimensions.

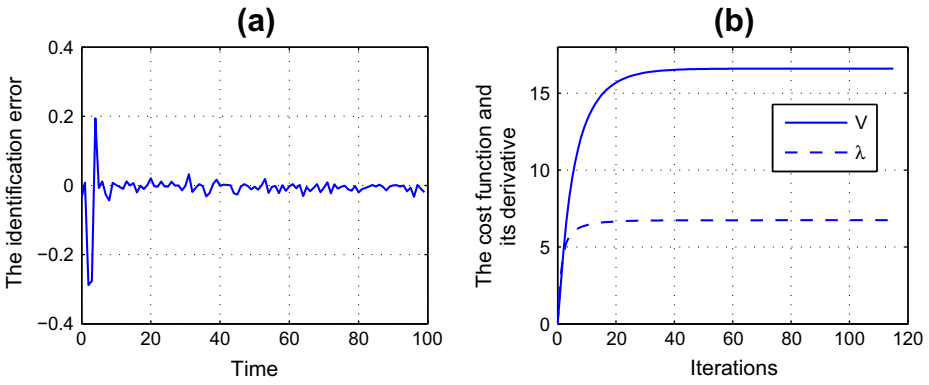

Fig. 3. (a) The system identification error. (b) The convergence process of the cost function and its derivative.

We choose three-layer feedforward NNs as model network, critic network, and action network with structures 2–8–1, 1–8–2, and 1–8–1, respectively. In the system identification process, the initial weights between input layer and hidden layer, and between hidden layer and output layer are chosen randomly in $[-0.5,0.5]$ and $[-0.1, 0.1]$, respectively. We apply the NN identification scheme for 100 time steps under the learning rate $\alpha_m = 0.05$ and obtain the result shown in Fig. 3a. We observe that the NN identifier successfully learns the unknown controlled system. Then, we finish the training of the model network and keep its weights unchanged.

Let the initial state $x_0 = 1.5$. Besides, the initial weights of the critic network and action network are all set to be random in $[-0.1, 0.1]$. Then, letting the adjusting parameter $\eta = 0.5$ and the learning rate $\alpha_c = \alpha_a = 0.05$, we train the critic network and action network for 115 iterations with each iteration of 2000 training epochs. The changing process of the cost function and its derivative of the iterative GDHP algorithm is shown in Fig. 3b, for $k = 0$, which displays the convergence of the two sequences.

For the purpose of making a comparison with the controller derived without considering the actuator saturation, we apply the controllers related to the two cases to system (34) for 20 time steps, respectively. The obtained simulation results are shown in Fig. 4. We can see that the restriction of actuator saturation has been overcome successfully. The excellent control performance verifies the effectiveness of the iterative GDHP algorithm.

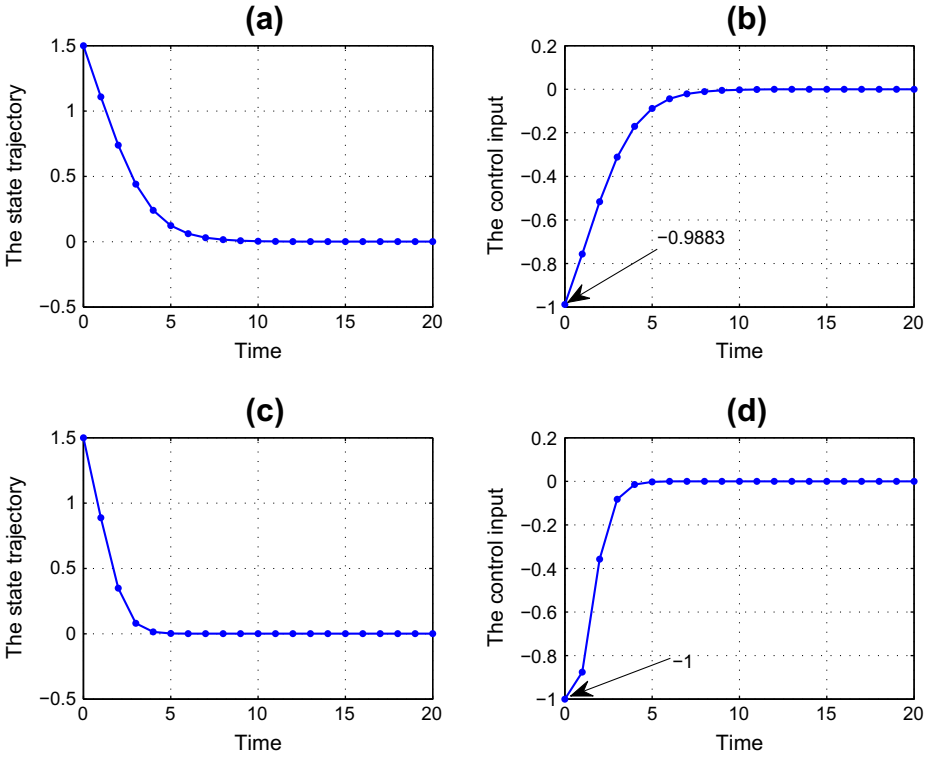

Fig. 4. Simulation results of Example 1. (a) The state trajectory x. (b) The control input u. (c) The state trajectory x without considering the control constraint. (d) The control input u without considering the control constraint.

Example 2. The following nonlinear system is a modification of the nonlinear equation in [15]:

$$x_{k+1} = \begin{bmatrix} x_{1k} + \sin(4u_k - 2x_{2k}) \\ x_{2k} - 2u_k \end{bmatrix}, \tag{36}$$

where $x_k = [x_{1k} x_{2k}]^T \in \mathbb{R}^2, u_k \in \mathbb{R}, k = 1, 2, \ldots$. We can see that $x_k = [0\ 0]^T$ is an equilibrium state of system (36). However, the system (36) is marginally stable at this equilibrium, since the eigenvalues of

$$\left.\frac{\partial x_{k+1}}{\partial x_k}\right|_{(0,0)} = \begin{bmatrix} 1 & -2 \\ 0 & 1 \end{bmatrix}$$

are all 1. It is desired to control the system with control constraint of $|u| \leqslant 0.5$. The cost function is chosen the same as in (35).

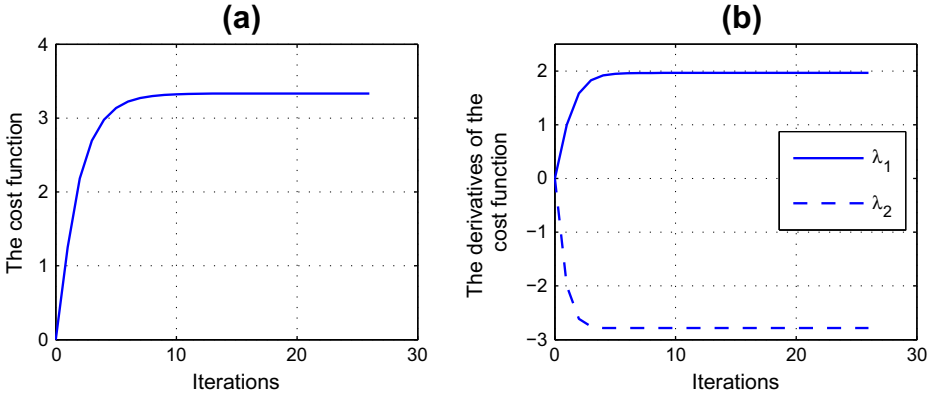

Fig. 5. (a) The convergence process of the cost function. (b) The convergence process of the derivatives of the cost function.

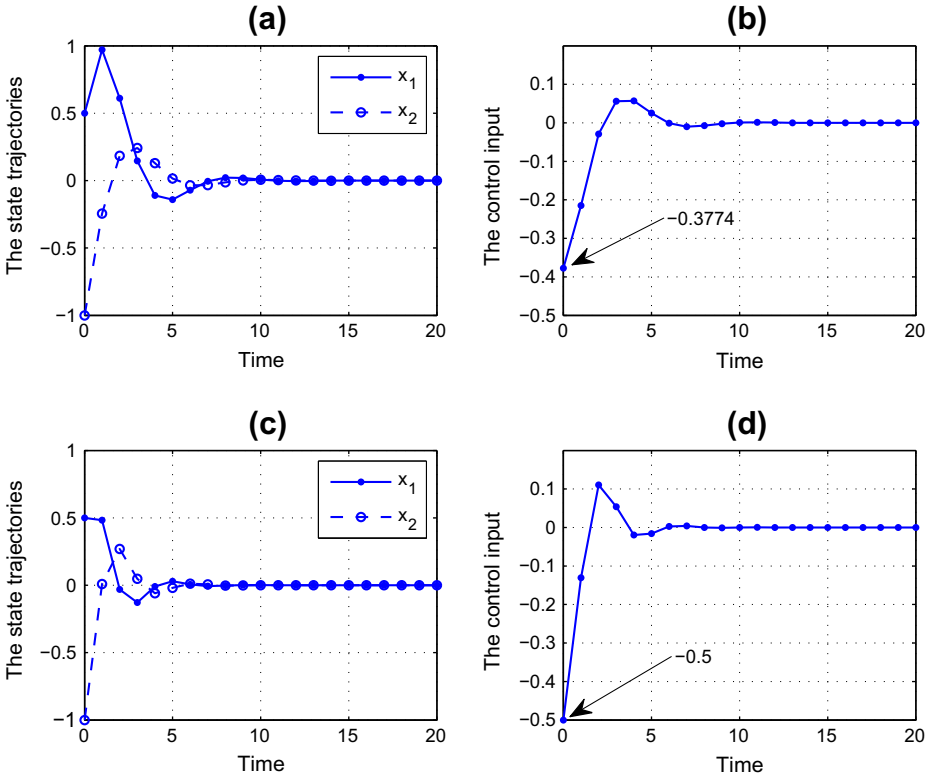

Fig. 6. Simulation results of Example 2. (a) The state trajectory x. (b) The control input u. (c) The state trajectory x without considering the control constraint. (d) The control input u without considering the control constraint.

In this example, the three NNs are chosen with structures 3–8–2, 2–8–3, and 2–8–1, respectively. Here, we train the critic network and action network for 26 iterations, while setting other parameters the same as in Example 1. When $k = 0$, the convergence process of the cost function and its derivatives is depicted in Fig. 5.

Next, for given initial state $[x_{10}x_{20}]^T = [0.5\ \ -1]^T$, we apply the optimal control laws designed by the iterative GDHP algorithm, with and without considering the actuator saturation, to system (36) for 20 time steps, respectively. The simulation results are shown in Fig. 6, which also exhibits excellent control effects of the iterative GDHP algorithm.

6. Conclusion

An iterative ADP algorithm is developed in this paper for near optimal control of unknown discrete-time nonlinear systems with control constraints. The GDHP technique is employed to perform the algorithm, with three NNs constructed to approximate the cost function and its derivatives, the control law, and the unknown controlled system, respectively. The numerical examples demonstrate the validity of the control scheme.

Since the tracking problem is another important topic of control engineering, it is necessary to expand the developed approach to solve the optimal tracking control problem in the future. Additionally, considering the fact that existing results about tracking control mainly aim at affine nonlinear systems, our future work will focus on dealing with the nonaffine case.

Acknowledgements

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] 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.
[2] A. Al-Tamimi, F.L. Lewis, M. Abu-Khalaf, Discrete-time nonlinear HJB solution using approximate dynamic programming: convergence proof, IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics 38 (2008) 943–949.
[3] R.A. Aliev, W. Pedrycz, B.G. Guirimov, R.R. Aliev, U. Ilhan, M. Babagil, S. Mammadli, Type-2 fuzzy neural networks with fuzzy clustering and differential evolution optimization, Information Sciences 181 (2011) 1591–1608.
[4] R.E. Bellman, Dynamic Programming, Princeton University Press, Princeton, NJ, 1957.
[5] D.P. Bertsekas, J.N. Tsitsiklis, Neuro-Dynamic Programming, Athena Scientific, Belmont, MA, 1996.
[6] D.P. Bertsekas, Temporal difference methods for general projected equations, IEEE Transactions on Automatic Control 56 (2011) 2128–2139.
[7] M.C. Campi, S.M. Savaresi, Direct nonlinear control design: the virtual reference feedback tuning (VRFT) approach, IEEE Transactions on Automatic Control 51 (2006) 14–27.
[8] J.I. Canelon, L.S. Shieh, N.B. Karayiannis, A new approach for neural control of nonlinear discrete dynamic systems, Information Sciences 174 (2005) 177–196.
[9] 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.
[10] J. Ding, S.N. Balakrishnan, Approximate dynamic programming solutions with a single network adaptive critic for a class of nonlinear systems, Journal of Control Theory and Applications 9 (2011) 370–380.
[11] H. Huang, H. Qin, Z. Hao, A. Lim, Example-based learning particle swarm optimization for continuous optimization, Information Sciences 182 (2012) 125–138.
[12] K.S. Hwang, H.Y. Lin, Y.P. Hsu, H.H. Yu, Self-organizing state aggregation for architecture design of Q-learning, Information Sciences 181 (2011) 2813–2822.
[13] S. Jagannathan, Neural Network Control of Nonlinear Discrete-Time Systems, CRC Press, Boca Raton, FL, 2006.
[14] J.H. Kim, F.L. Lewis, Model-free H_∞ control design for unknown linear discrete-time systems via Q-learning with LMI, Automatica 46 (2010) 1320–1326.
[15] A.U. Levin, K.S. Narendra, Control of nonlinear dynamical systems using neural networks: controllability and stabilization, IEEE Transactions on Neural Networks 4 (1993) 192–206.
[16] F.L. Lewis, V.L. Syrmos, Optimal Control, Wiley, New York, 1995.
[17] F.L. Lewis, D. Vrabie, Reinforcement learning and adaptive dynamic programming for feedback control, IEEE Circuits and Systems Magazine 9 (2009) 32–50.
[18] C.K. Lin, Robust adaptive critic control of nonlinear systems using fuzzy basis function networks: an LMI approach, Information Sciences 177 (2007) 4934–4946.
[19] Z. Man, K. Lee, D. Wang, Z. Cao, C. Miao, A new robust training algorithm for a class of single-hidden layer feedforward neural networks, Neurocomputing 74 (2011) 2491–2501.
[20] S. Preitl, R.E. Precup, J. Fodor, B. Bede, Iterative feedback tuning in fuzzy control systems: theory and applications, Acta Polytechnica Hungarica 3 (2006) 81–96.
[21] D.V. Prokhorov, D.C. Wunsch, Adaptive critic designs, IEEE Transactions on Neural Networks 8 (1997) 997–1007.
[22] M.B. Radac, R.E. Precup, E.M. Petriu, S. Preitl, Application of IFT and SPSA to servo system control, IEEE Transactions on Neural Networks 22 (2011) 2363–2375.
[23] J. Si, Y.T. Wang, On-line learning control by association and reinforcement, IEEE Transactions on Neural Networks 12 (2001) 264–276.
[24] 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.
[25] K.G. Vamvoudakis, F.L. Lewis, Online actor-critic algorithm to solve the continuous-time infinite horizon optimal control problem, Automatica 46 (2010) 878–888.
[26] 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 Transactions on Neural Networks 13 (2002) 764–773.
[27] 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.

[28] D. Wang, D. Liu, Optimal control for a class of unknown nonlinear systems via the iterative GDHP algorithm, in: Proceedings of 8th International Symposium on Neural Networks, Guilin, China, 2011, pp. 630–639.
[29] D. Wang, D. Liu, Q. Wei, Adaptive dynamic programming for finite-horizon optimal tracking control of a class of nonlinear systems, in: Proceedings of the 30th Chinese Control Conference, Yantai, China, 2011, pp. 2450–2455.
[30] F.Y. Wang, H. Zhang, D. Liu, Adaptive dynamic programming: an introduction, IEEE Computational Intelligence Magazine 4 (2009) 39–47.
[31] F.Y. Wang, N. Jin, D. Liu, Q. Wei, Adaptive dynamic programming for finite-horizon optimal control of discrete-time nonlinear systems with ε-error bound, IEEE Transactions on Neural Networks 22 (2011) 24–36.
[32] X.S. Wang, Y.H. Cheng, J.Q. Yi, A fuzzy actor-critic reinforcement learning network, Information Sciences 177 (2007) 3764–3781.
[33] P.J. Werbos, Advanced forecasting methods for global crisis warning and models of intelligence, General Systems Yearbook 22 (1977) 25–38.
[34] 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: Neural, Fuzzy, and Adaptive Approaches, Van Nostrand Reinhold, New York, 1992.
[35] W. Wu, L. Li, J. Yang, Y. Liu, A modified gradient-based neuro-fuzzy learning algorithm and its convergence, Information Sciences 180 (2010) 1630–1642.
[36] 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 Transactions on Neural Networks 20 (2009) 1490–1503.
[37] R. Zhang, Z.B. Xu, G.B. Huang, D. Wang, Global convergence of online BP training with dynamic learning rate, IEEE Transactions on Neural Networks and Learning Systems 23 (2012) 330–341.