Neural-Network-Based Near-Optimal Control for a Class of Discrete-Time Affine Nonlinear Systems With Control Constraints

Huaguang Zhang, *Senior Member, IEEE*, Yanhong Luo, *Member, IEEE*, and Derong Liu, *Fellow, IEEE*

***Abstract*—In this paper, the near-optimal control problem for a class of nonlinear discrete-time systems with control constraints is solved by iterative adaptive dynamic programming algorithm. First, a novel nonquadratic performance functional is introduced to overcome the control constraints, and then an iterative adaptive dynamic programming algorithm is developed to solve the optimal feedback control problem of the original constrained system with convergence analysis. In the present control scheme, there are three neural networks used as parametric structures for facilitating the implementation of the iterative algorithm. Two examples are given to demonstrate the convergence and feasibility of the proposed optimal control scheme.**

***Index Terms*—Adaptive dynamic programming, approximate dynamic programming, control constraints, convergence analysis, near-optimal control, neural networks.**

I. Introduction

SATURATION, dead zone, backlash, and hysteresis are the most common actuator nonlinearities in practical control system applications. Saturation nonlinearity is unavoidable in most actuators. When an actuator has reached certain input limit, it is said to be "saturated," since efforts to further increase the actuator output would not result in any variation in the output. Due to the nonanalytic nature of the actuator nonlinear dynamics and the fact that the exact actuator nonlinear functions are unknown, such systems present a challenge to control engineers. The control of systems with saturating actuators has been the focus of many researchers for many years. Several methods for deriving control laws considering the saturation phenomena can be found in [10], [33], and [38]. However, most of these methods do not consider optimal control laws for general nonlinear discrete-time systems. In this paper, we study this problem through the framework of the Hamilton–Jacobi–Bellman (HJB) equation from optimal control theory [6], [9]. Solving the HJB equation is a challenging problem due to its inherently nonlinear nature. For nonlinear systems, the HJB equation generally cannot be solved analytically. If the system's inputs are constrained, the derived HJB equation is more difficult to solve and often becomes unsolvable.

For systems with input constraints, a nonquadratic functional is proposed in [26]–[28] to confront this kind of constraints. Using nonquadratic functionals, the HJB equation is formulated and its solution results in a smooth saturated controller. However it remains difficult to actually solve for the optimal value function of this HJB equation.

In recent years, in order to obtain approximate solutions of the HJB equation, adaptive/approximate dynamic programming (ADP) algorithms have gained much attention from researchers (cf., [1]–[4], [7], [8], [12]–[25], [29], [32], [34]–[37], [39], [42], [43], and [45]–[47]). ADP was proposed in [11], [40], [41], and [44] as a way for solving optimal control problems forward in time. In [42], ADP approaches were classified into several schemes including heuristic dynamic programming (HDP), action-dependent heuristic dynamic programming (ADHDP), also known as Q-learning [40], dual heuristic dynamic programming (DHP), ADDHP, globalized DHP (GDHP), and ADGDHP. Moreover, in [3], a greedy iterative HDP scheme with convergence proof is proposed for solving the optimal control problem for nonlinear discrete-time systems with known mathematical model, which does not require an initially stable policy. In [12], a successive approximation method using generalized Hamilton–Jacobi–Bellman (GHJB) equation is proposed to solve the near-optimal control problem for affine nonlinear discrete-time systems, which requires small perturbation assumption and an initially stable policy.

Though ADP algorithms have made great progress in the optimal control field, to the best of our knowledge, it is still an open problem how to solve the optimal control problem for discrete-time systems with control constraints based on ADP algorithms. In this paper, we will give a positive answer to the question on how to find a constrained optimal control if the actuator has saturating characteristic. First, the HJB equation for discrete-time systems with control constraints is derived using nonquadratic functionals. In order to solve this HJB equation, a new iterative adaptive dynamic programming algorithm is presented with convergence proof. Furthermore, in order to facilitate the implementation of the iterative ADP algorithm, we show how to

Manuscript received August 01, 2008; revised December 07, 2008 and March 04, 2009; accepted April 05, 2009. First published August 04, 2009; current version published September 02, 2009. This work was supported by the National Natural Science Foundation of China under Grants 60534010, 60774048, 60728307, 60521003, and 60621001, the Program for Cheung Kong Scholars, the Research Fund for the Doctoral Program of China Higher Education (20070145015), the 111 Project of Education Ministry of China (B08015), and the National Basic Research Program of China (2009CB320601).

H. Zhang and Y. Luo are with the School of Information Science and Engineering, Northeastern University, Shenyang, Liaoning 110004, China (e-mail: hgzhang@ieee.org; neuluo@163.com).

D. Liu is with the Key Laboratory of Complex Systems and Intelligence Science, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China (e-mail: derong.liu@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/TNN.2009.2027233

introduce neural networks to obtain the costate function. This in turn results in a near-optimal state feedback controller suitable for saturated actuators. Moreover, for the nonlinear plant whose mathematical model is too difficult to construct, we employ a model network to approximate the nonlinear dynamics of the plant.

In brief, the main contributions of this paper are as follows.

1) Propose a novel nonquadratic functional to deal with control constraints of nonlinear discrete-time systems and derive the corresponding discrete-time HJB equation.
2) Prove that the iterative value function sequence converges to the optimal value function, i.e., the infimum of all the value functions obtained by admissible control law sequences, and show that this optimal value function satisfies the HJB equation.
3) Implement the iterative adaptive dynamic programming algorithm by introducing the costate function, which avoids the computation of a derivative term and an integral term in solving the optimal control law.
4) Utilize two neural networks to approximate the costate function and the corresponding control law, respectively. Specifically, use a model network to approximate the nonlinear system dynamics, which renders the iterative adaptive dynamic programming algorithm suitable to the plants whose mathematical models are unknown.

This paper is organized as follows. In Section II, the discrete-time HJB equation is presented for constrained nonlinear systems. In Section III, the optimal control scheme is developed based on iterative adaptive dynamic programming algorithm and the corresponding neural network implementation of the iterative algorithm is presented. In Section IV, two examples are given to demonstrate the effectiveness of the proposed optimal control scheme. In Section V, concluding remarks are given.

II. Discrete-Time HJB Equation for Constrained Nonlinear Systems

Consider a class of discrete-time affine nonlinear systems as follows:

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

where $x(k) \in \Re^n$ is the state vector, and $f: \Re^n \to \Re^n$ and $g: \Re^n \to \Re^{n\times m}$ are differentiable with respect to their arguments with $f(0) = 0$. Assume that $f + gu$ is Lipschitz continuous on a set Ω in $\Re^n$ containing the origin, and that the system (1) is controllable in the sense that there exists at least a continuous control law on Ω that asymptotically stabilizes the system. We denote $\Omega_u = \{u(k) \mid u(k) = [u_1(k), u_2(k), \ldots, u_m(k)]^T \in \Re^m, |u_i(k)| \le \bar{u}_i, i = 1, \ldots, m\}$, where $\bar{u}_i$ is the saturating bound for the ith actuator. Let $\bar{U} \in \Re^{m\times m}$ be the constant diagonal matrix given by $\bar{U} = \mathrm{diag}[\bar{u}_1, \bar{u}_2, \ldots, \bar{u}_m]$.

In this paper, we study the design of an optimal controller for this class of constrained discrete-time systems, i.e., our objective is to find an optimal state feedback controller that can make $x(k) \to 0$ when $k \to \infty$. Specifically, it is desired to find the optimal control law $v(x)$ so that the control sequence $u(\cdot) = (u(k), u(k+1), \ldots)$ with each $u(i) \in \Omega_u$, $i = k, k+1, \ldots$, minimizes the generalized performance functional as follows:

$$J(x(k), u(\cdot)) = \sum_{i=k}^{\infty} \left\{ x(i)^T Q x(i) + W(u(i)) \right\} \tag{2}$$

where $u(i) = v(x(i))$, $W(u(i)) \in \Re$ is positive definite, and the weight matrix Q is also positive definite.

For optimal control problems, the state feedback control law $v(x)$ must not only stabilize the system on Ω but guarantee that (2) is finite. Such a control law is said to be admissible [8].

Definition 1: A control law $v(x)$ is said to be admissible with respect to (2) on Ω if $v(x)$ is continuous with $v(x(k)) \in \Omega_u$ for $\forall x(k) \in \Omega$ and stabilizes (1) on Ω, $v(0) = 0$, and for $\forall x(0) \in \Omega$, $J(x(0), u(\cdot))$ is finite, where $u(\cdot) = (u(0), u(1), \ldots)$ and $u(k) = v(x(k))$, $k = 0, 1, \ldots$.

Based on the above definition, we are ready to explain the admissible control law sequence. A control law sequence $\{\eta_i\} = (\eta_\infty, \ldots, \eta_1, \eta_0)$ is called admissible if the resultant control sequence $(u(0), u(1), \ldots, u(\infty))$ stabilizes the system (1) with any initial state $x(0)$ and guarantees that $J(x(0), u(\cdot))$ is finite. It should be mentioned that in this case, each control action obeys a different control law, i.e., $u(i)$ is produced by control law $\eta_{\infty-i}$ for $i = 0, 1, \ldots$. Moreover, the control law sequence $\{\eta_i\} = (\eta_\infty, \ldots, \eta_1, \eta_0)$ is also called the nonstationary policy in the literature [5].

For convenience, in the sequel, $V^*(x(k))$ is used to denote the optimal value function which is defined as $V^*(x(k)) = \min_{u(\cdot)} J(x(k), u(\cdot))$, and $v^*(x)$ is used to denote the corresponding optimal control law.

For unconstrained control problem, $W(u(i))$ in the performance functional (2) is commonly chosen as the quadratic form of the control input $u(i)$. However, in this paper, to confront the bounded control problem, based on [28], we employ a nonquadratic functional as follows:

$$\begin{aligned} W(u(i)) &= 2\int_0^{u(i)} \boldsymbol{\varphi}^{-T}(\bar{U}^{-1}s)\bar{U}R\,ds \\ \boldsymbol{\varphi}^{-1}(u(i)) &= [\varphi^{-1}(u_1(i)), \varphi^{-1}(u_2(i)), \ldots, \varphi^{-1}(u_m(i))]^T \end{aligned} \tag{3}$$

where R is positive definite and assumed to be diagonal for simplicity of analysis, $s \in \Re^m$, $\boldsymbol{\varphi} \in \Re^m$, $\boldsymbol{\varphi}^{-T}$ denotes $\left(\boldsymbol{\varphi}^{-1}\right)^T$, and $\varphi(\cdot)$ is a bounded one-to-one function satisfying $|\varphi(\cdot)| \le 1$ and belonging to C^p $(p \ge 1)$ and $L_2(\Omega)$. Moreover, it is a monotonic increasing odd function with its first derivative bounded by a constant M. Such a function is easy to find, and one example is the hyperbolic tangent function $\varphi(\cdot) = \tanh(\cdot)$. It should be noticed that by the definition above, $W(u(i))$ is assured to be positive definite because $\varphi^{-1}(\cdot)$ is a monotonic odd function and R is positive definite.

According to Bellman's principle of optimality, the optimal value function $V^*(x)$ should satisfy the following HJB equation:

$$\begin{aligned} V^*(x(k)) &= \min_{u(\cdot)} \sum_{i=k}^{\infty} \left\{ x(i)^T Q x(i) + 2\int_0^{u(i)} \boldsymbol{\varphi}^{-T}(\bar{U}^{-1}s)\bar{U}R\,ds \right\} \\ &= \min_{u(k)} \left\{ x(k)^T Q x(k) + 2\int_0^{u(k)} \boldsymbol{\varphi}^{-T}(\bar{U}^{-1}s)\bar{U}R\,ds \right. \\ &\qquad \left. + V^*(x(k+1)) \right\}. \end{aligned} \tag{4}$$

The optimal control law $v^*(x)$ should satisfy

$$v^*(x(k))=\arg\min_{u(k)}\left\{x(k)^TQx(k)+2\int_0^{u(k)}\varphi^{-T}(\bar{U}^{-1}s)\bar{U}Rds + V^*(x(k+1))\right\}. \quad (5)$$

The optimal control problem can be solved if the optimal value function $V^*(x)$ can be obtained from (4). However, there is currently no method for solving this value function of the constrained optimal control problem. Therefore, in the next section, we will discuss how to utilize the iterative adaptive dynamic programming algorithm to seek the near-optimal control solution.

III. The Near-Optimal Control Based on Iterative Adaptive Dynamic Programming Algorithm

This section consists of three subsections. In the first subsection, the iterative adaptive dynamic programming algorithm is derived, while the corresponding convergence proof is presented in the second subsection, and the implementation of the optimal control scheme is described in the third subsection.

A. Derivation of the Iterative Adaptive Dynamic Programming Algorithm

Since direct solution of the HJB equation is computationally intensive, we develop in this paper an iterative adaptive dynamic programming algorithm, based on Bellman's principle of optimality and the greedy iteration principle.

First, we start with initial value function $V_0(\cdot)=0$ which is not necessarily the optimal value function. Then, we find the law of single control vector $v_0(x)$ as follows:

$$v_0(x(k))=\arg\min_{u(k)}\left\{x(k)^TQx(k)+2\int_0^{u(k)}\varphi^{-T}(\bar{U}^{-1}s)\bar{U}Rds + V_0(x(k+1))\right\} \quad (6)$$

and update the value function as

$$V_1(x(k))=x(k)^TQx(k)+2\int_0^{v_0(x(k))}\varphi^{-T}(\bar{U}^{-1}s)\bar{U}Rds. \quad (7)$$

Therefore, for $i=1,2,\ldots$, the iterative adaptive dynamic programming algorithm then iterates between

$$v_i(x(k))=\arg\min_{u(k)}\left\{x(k)^TQx(k)+2\int_0^{u(k)}\varphi^{-T}(\bar{U}^{-1}s)\bar{U}Rds + V_i(x(k+1))\right\} \quad (8)$$

and

$$V_{i+1}(x(k))=\min_{u(k)}\left\{x(k)^TQx(k)+2\int_0^{u(k)}\varphi^{-T}(\bar{U}^{-1}s)\bar{U}Rds + V_i(x(k+1))\right\}. \quad (9)$$

It can be seen that based on (8), (9) can further be written as

$$V_{i+1}(x(k))=x(k)^TQx(k)+2\int_0^{v_i(x(k))}\varphi^{-T}(\bar{U}^{-1}s)\bar{U}Rds + V_i(x(k+1)) \quad (10)$$

where $x(k+1)=f(x(k))+g(x(k))v_i(x(k))$.

In summary, in this iterative algorithm, the value function sequence $\{V_i\}$ and control law sequence $\{v_i\}$ are updated by implementing the iteration between (8) and (10) with the iteration number i increasing from 0 to ∞.

To further explain the iteration process, next we analyze this iterative algorithm. First, based on (10), we can obtain

$$V_i(x(k+1))=x(k+1)^TQx(k+1) + 2\int_0^{v_{i-1}(x(k+1))}\varphi^{-T}(\bar{U}^{-1}s)\bar{U}Rds + V_{i-1}(x(k+2)) \quad (11)$$

where $x(k+2)=f(x(k+1))+g(x(k+1))v_{i-1}(x(k+1))$. Then, by further expanding (10), we have

$$\begin{aligned}V_{i+1}(x(k)) &= x(k)^TQx(k)+2\int_0^{v_i(x(k))}\varphi^{-T}(\bar{U}^{-1}s)\bar{U}Rds \\ &\quad + x(k+1)^TQx(k+1) \\ &\quad + 2\int_0^{v_{i-1}(x(k+1))}\varphi^{-T}(\bar{U}^{-1}s)\bar{U}Rds \\ &\quad + \cdots + x(k+i)^TQx(k+i) \\ &\quad + 2\int_0^{v_0(x(k+i))}\varphi^{-T}(\bar{U}^{-1}s)\bar{U}Rds \\ &\quad + V_0(x(k+i+1))\end{aligned} \quad (12)$$

where $V_0(x(k+i+1))=0$.

From (12), it can be seen that during the iteration process, the control actions for different control steps obey different control laws. After the iteration number $i+1$, the obtained control law sequence is $(v_i, v_{i-1}, \ldots, v_0)$. With the iteration number i increasing to ∞, the obtained control law sequence has the length of ∞. For the infinite-horizon problem, both the optimal value function and the optimal control law are unique. Therefore, it is desired that the control law sequence will converge when the iteration number $i\to\infty$. In the following, we will prove that both the value function sequence $\{V_i\}$ and the control law sequence $\{v_i\}$ are convergent.

B. Convergence Analysis of the Iterative Adaptive Dynamic Programming Algorithm

For the unconstrained case, a convergence proof of the value-iteration-based HDP algorithm is addressed in [4]. In this paper, in order to prove the convergence characteristics of the iterative ADP algorithm for constrained nonlinear systems, we first present two lemmas before presenting our theorems. For convenience, the nonquadratic functional $2\int_0^{u(k)}\varphi^{-T}(\bar{U}^{-1}s)\bar{U}Rds$ will be written as $W(u(k))$ in the sequel.

Lemma 1: Let $\{\mu_i\}$ be an arbitrary sequence of control laws, and $\{v_i\}$ be the control law sequence as in (8). Let V_i be as in (9) and Λ_i be

$$\Lambda_{i+1}(x(k))=x(k)^TQx(k)+W(\mu_i(x(k)))+\Lambda_i(x(k+1)). \quad (13)$$

If $V_0(\cdot)=\Lambda_0(\cdot)=0$, then $V_i(x)\le\Lambda_i(x), \forall i$.

Proof: It is clear from the fact that V_{i+1} is the result of minimizing the right-hand side of (9) with respect to the control input $u(k)$, while Λ_{i+1} is a result of arbitrary control input. ■

Lemma 2: Let the sequence $\{V_i\}$ be defined as in (9). If the system is controllable, then there is an upper bound Y such that $0 \leq V_i(x(k)) \leq Y, \forall i$.

Proof: Let $\{\eta_i(x)\}$ be a sequence of stabilizing and admissible control laws, and let $V_0(\cdot) = P_0(\cdot) = 0$, where V_i is updated by (9) and P_i is updated by

$$P_{i+1}(x(k)) = x(k)^T Q x(k) + W(\eta_i(x(k))) + P_i(x(k+1)). \tag{14}$$

From (14), we can further obtain

$$\begin{aligned} P_i(x(k+1)) = {} & x(k+1)^T Q x(k+1) \\ & + W(\eta_{i-1}(x(k+1))) + P_{i-1}(x(k+2)). \end{aligned} \tag{15}$$

Thus, the following relation can be obtained:

$$\begin{aligned} & P_{i+1}(x(k)) \\ & = x(k)^T Q x(k) + W(\eta_i(x(k))) + x(k+1)^T Q x(k+1) \\ & \quad + W(\eta_{i-1}(x(k+1))) + P_{i-1}(x(k+2)) \\ & = x(k)^T Q x(k) + W(\eta_i(x(k))) + x(k+1)^T Q x(k+1) \\ & \quad + W(\eta_{i-1}(x(k+1))) + x(k+2)^T Q x(k+2) \\ & \quad + W(\eta_{i-2}(x(k+2))) + P_{i-2}(x(k+3)) \\ & \quad \vdots \\ & = x(k)^T Q x(k) + W(\eta_i(x(k))) + x(k+1)^T Q x(k+1) \\ & \quad + W(\eta_{i-1}(x(k+1))) + x(k+2)^T Q x(k+2) \\ & \quad + W(\eta_{i-2}(x(k+2))) \\ & \quad + \cdots \\ & \quad + x(k+i)^T Q x(k+i) + W(\eta_0(x(k+i))) \\ & \quad + P_0(x(k+i+1)) \end{aligned} \tag{16}$$

where $P_0(x(k+i+1)) = 0$.

Let $U_i(x(k)) = x(k)^T Q x(k) + W(\eta_i(x(k)))$, and then (16) can further be written as

$$\begin{aligned} & P_{i+1}(x(k)) \\ & = \sum_{j=0}^{i} U_{i-j}(x(k+j)) \\ & = \sum_{j=0}^{i} \Big\{ x(k+j)^T Q x(k+j) + W(\eta_{i-j}(x(k+j))) \Big\} \\ & \leq \lim_{i \to \infty} \sum_{j=0}^{i} \Big\{ x(k+j)^T Q x(k+j) + W\big(\eta_{i-j}(x(k+j))\big) \Big\}. \end{aligned} \tag{17}$$

Note that $\{\eta_i(x)\}$ is an admissible control law sequence, i.e., $x(k) \to 0$ as $k \to \infty$. Therefore, there exists an upper bound Y such that

$$\forall i: \ P_{i+1}(x(k)) \leq \lim_{i \to \infty} \sum_{j=0}^{i} U_{i-j}(x(k+j)) \leq Y. \tag{18}$$

Combining with Lemma 1, we can obtain

$$\forall i: \ V_{i+1}(x(k)) \leq P_{i+1}(x(k)) \leq Y. \tag{19}$$

■

Next, Lemmas 1 and 2 will be used in the proof of our main theorems.

Theorem 1: Define the value function sequence $\{V_i\}$ as in (10) with $V_0(\cdot) = 0$, and the control law sequence $\{v_i\}$ as in (8). Then, we can conclude that $\{V_i\}$ is a nondecreasing sequence satisfying $V_{i+1}(x(k)) \geq V_i(x(k))$, $\forall i$.

Proof: For convenience of analysis, define a new sequence $\{\Phi_i\}$ as follows:

$$\Phi_{i+1}(x(k)) = x(k)^T Q x(k) + W(v_{i+1}(x(k))) + \Phi_i(x(k+1)) \tag{20}$$

with $\Phi_0(\cdot) = V_0(\cdot) = 0$. The control law sequence $\{v_i\}$ is updated by (8) and the value function sequence $\{V_i\}$ is updated by (10).

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

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

$$V_1(x(k)) - \Phi_0(x(k)) = x(k)^T Q x(k) + W(v_0(x(k))) \geq 0 \tag{21}$$

we have for $i = 0$

$$V_1(x(k)) \geq \Phi_0(x(k)). \tag{22}$$

Second, we assume that it holds for $i-1$, i.e., for any $x(k)$, $V_i(x(k)) \geq \Phi_{i-1}(x(k))$. Then, for i, since

$$\Phi_i(x(k)) = x(k)^T Q x(k) + W(v_i(x(k))) + \Phi_{i-1}(x(k+1)) \tag{23}$$

and

$$V_{i+1}(x(k)) = x(k)^T Q x(k) + W(v_i(x(k))) + V_i(x(k+1)) \tag{24}$$

hold, we can obtain

$$V_{i+1}(x(k)) - \Phi_i(x(k)) = V_i(x(k+1)) - \Phi_{i-1}(x(k+1)) \geq 0 \tag{25}$$

i.e., the following equation holds:

$$\Phi_i(x(k)) \leq V_{i+1}(x(k)). \tag{26}$$

Therefore, (26) is proved $\forall i$ by mathematical induction.

Furthermore, from Lemma 1, we know that $V_i(x(k)) \leq \Phi_i(x(k))$. Therefore, we have

$$V_i(x(k)) \leq \Phi_i(x(k)) \leq V_{i+1}(x(k)). \tag{27}$$

■

Next, we are ready to exploit the limit of the value function sequence $\{V_i\}$ when $i \to \infty$.

Let $\left\{\eta_i^{(l)}\right\}$ be the lth admissible control law sequence. Similar to the proof of Lemma 2, we can construct the associated sequence $P_i^{(l)}(x)$ as follows:

$$P_{i+1}^{(l)}(x(k)) = x(k)^T Q x(k) + W(\eta_i^{(l)}(x(k))) + P_i^{(l)}(x(k+1)) \tag{28}$$

with $P_0^{(l)}(\cdot) = 0$.

Let $U_i^{(l)}(x(k)) = x(k)^T Q x(k) + W(\eta_i^{(l)}(x(k)))$. Then, the following relation can be obtained:

$$P_{i+1}^{(l)}(x(k)) = \sum_{j=0}^{i} U_{i-j}^{(l)}(x(k+j)). \tag{29}$$

Let $i \to \infty$, then we have

$$P_{\infty}^{(l)}(x(k)) = \lim_{i\to\infty} \sum_{j=0}^{i} U_{i-j}^{(l)}(x(k+j)). \tag{30}$$

Combining (29) and (30), we can obtain

$$P_{i+1}^{(l)}(x(k)) \le P_{\infty}^{(l)}(x(k)). \tag{31}$$

Theorem 2: Define $P_{\infty}^{(l)}(x(k))$ as in (30), and the value function sequence $\{V_i\}$ as in (10) with $V_0(\cdot) = 0$. For any state vector $x(k)$, define $V^*(x(k)) = \inf_l\{P_{\infty}^{(l)}(x(k))\}$, which can be considered as the "optimal" value function starting from $x(k)$ under all admissible control law sequences with length ∞. Then, we can conclude that V^* is the limit of the value function sequence $\{V_i\}$.

Proof: According to the definition of $P_{\infty}^{(l)}(x(k))$, the associated control law sequence $\{\eta_i^{(l)}(x)$ is admissible. Thus, it is guaranteed that $\lim_{i\to\infty}\sum_{j=0}^{i} U_{i-j}^{(l)}(x(k+j))$ is finite, i.e., $P_{\infty}^{(l)}(x(k))$ is finite. Hence, for any l, there exists an upper bound Y_l such that

$$P_{i+1}^{(l)}(x(k)) \le P_{\infty}^{(l)}(x(k)) \le Y_l. \tag{32}$$

Combining with Lemma 1, we can further obtain

$$\forall\, l, i: \quad V_{i+1}(x(k)) \le P_{i+1}^{(l)}(x(k)) \le Y_l. \tag{33}$$

Since $V^*(x(k)) = \inf_l\{P_{\infty}^{(l)}(x(k))\}$, for any $\epsilon > 0$, there exists a sequence of admissible control laws $\{\eta_i^{(K)}\}$ such that the associated value function satisfies $P_{\infty}^{(K)}(x(k)) \le V^*(x(k)) + \epsilon$. According to (33), we have $V_i(x(k)) \le P_i^{(l)}(x(k))$ for any l and i. Thus, we can obtain $\lim_{i\to\infty} V_i(x(k)) \le P_{\infty}^{(K)}(x(k)) \le V^*(x(k)) + \epsilon$. Noting that ϵ is chosen arbitrarily, we have

$$\lim_{i\to\infty} V_i(x(k)) \le V^*(x(k)). \tag{34}$$

On the other hand, since $V_{i+1}(x(k)) \le P_{i+1}^{(l)}(x(k)) \le Y_l$, $\forall l, i$, we have $\lim_{i\to\infty} V_i(x(k)) \le \inf_l\{Y_l\}$. According to the definition of admissible control law sequence, the control law sequence associated with the value function $\lim_{i\to\infty} V_i(x(k))$ must be an admissible control law sequence, i.e., there exists a sequence of admissible control laws $\{\eta_i^{(N)}\}$ such that $\lim_{i\to\infty} V_i(x(k)) = P_{\infty}^{(N)}(x(k))$. Combining with the definition $V^*(x(k)) = \inf_l\{P_{\infty}^{(l)}(x(k))\}$, we can obtain

$$\lim_{i\to\infty} V_i(x(k)) \ge V^*(x(k)). \tag{35}$$

Therefore, combining (34) and (35), we can conclude that $\lim_{i\to\infty} V_i(x(k)) = V^*(x(k))$, i.e., V^* is the limit of the value function sequence $\{V_i\}$. ■

Next, let us consider what will happen when we make $i \to \infty$ in (9). The left-hand side is simply $V_{\infty}(x)$. But for the right-hand side, it is not obvious to see since the minimum will be reached at different $u(k)$ for different i. However, the following result can be proved.

Theorem 3: For any state vector $x(k)$, the "optimal" value function $V^*(x)$ satisfies the HJB equation $V^*(x(k)) = \inf_{u(k)}\{x(k)^T Q x(k) + W(u(k)) + V^*(x(k+1))\}$.

Proof: For any $u(k)$ and i, according to (9), we have

$$V_i(x(k)) \le x(k)^T Q x(k) + W(u(k)) + V_{i-1}(x(k+1)). \tag{36}$$

According to Theorems 1 and 2, the value function sequence $\{V_i\}$ is a nondecreasing sequence satisfying $\lim_{i\to\infty} V_i(x(k)) = V^*(x(k))$, hence the relation $V_{i-1}(x(k+1)) \le V^*(x(k+1))$ holds for any i. Thus, we can obtain

$$V_i(x(k)) \le x(k)^T Q x(k) + W(u(k)) + V^*(x(k+1)). \tag{37}$$

Let $i \to \infty$, then we have

$$V^*(x(k)) \le x(k)^T Q x(k) + W(u(k)) + V^*(x(k+1)). \tag{38}$$

Since $u(k)$ in the above equation is chosen arbitrarily, the following equation holds:

$$V^*(x(k)) \le \inf_{u(k)} \left\{x(k)^T Q x(k) + W(u(k)) + V^*(x(k+1))\right\}. \tag{39}$$

On the other hand, for any i, the value function sequence satisfies

$$V_i(x(k)) = \min_{u(k)} \left\{x(k)^T Q x(k) + W(u(k)) + V_{i-1}(x(k+1))\right\}. \tag{40}$$

Combining with $V_i(x(k)) \le V^*(x(k))$, $\forall i$, we have

$$V^*(x(k)) \ge \inf_{u(k)} \left\{x(k)^T Q x(k) + W(u(k)) + V_{i-1}(x(k+1))\right\}. \tag{41}$$

Let $i \to \infty$, and then we can obtain

$$V^*(x(k)) \ge \inf_{u(k)} \left\{x(k)^T Q x(k) + W(u(k)) + V^*(x(k+1))\right\}. \tag{42}$$

Combining (39) and (42), we have

$$V^*(x(k)) = \inf_{u(k)} \{x(k)^T Q x(k) + W(u(k)) + V^*(x(k+1))\}. \tag{43}$$

■

According to Theorems 1 and 2, we can conclude that $V_i(x(k)) \le V_{i+1}(x(k))$, $\forall i$ and $\lim_{i\to\infty} V_i(x(k)) = V^*(x(k))$. Furthermore, according to Theorem 3, we have $V^*(x(k)) = \inf_{u(k)}\{x(k)^T Q x(k) + W(u(k)) + V^*(x(k+1))\}$. Therefore, we can conclude that the value function sequence $\{V_i\}$ converges to the optimal value function of the discrete-time HJB equation, i.e., $V_i \to V^*$ as $i \to \infty$. Since the value function sequence is convergent, according to (5) and (8),

we can conclude that the corresponding control law sequence $\{v_i\}$ converges to the optimal control law v^* as $i \to \infty$.

It should be mentioned that the value function $V_i(x)$ we constructed is a new function that is different from ordinary performance functional. Via Lemma 2 and Theorem 1, we have shown that for any $x(k) \in \Omega$, the function sequence $\{V_i(x(k))\}$ is a nondecreasing sequence, which will increase its value with an upper bound. This is in contrast to other works in the literature, e.g., [12], where the value functions are constructed as a nonincreasing sequence with lower bound. Moreover, it should be noted that we do not require every *control law* in the sequence $\{v_i\}$ to be admissible. What we need is a *control law sequence* to be admissible, i.e., the resultant *sequence* of control vectors to stabilize the system. Similar framework can be found in [4].

C. Implementation of the Iterative Adaptive Dynamic Programming Algorithm

1) Derivation of the Iterative Dual Heuristic Programming (DHP) Algorithm: First, we assume that the value function $V_i(x)$ is smooth. In order to implement the iteration between (8) and (10), for $i = 0, 1, \ldots$, we further assume that the minimum of the right-hand side of (8) can exactly be solved by letting the gradient of the right-hand side of (8) with respect to $u(k)$ equal to zero, i.e.,

$$\frac{\partial \left(x(k)^T Q x(k) + W(u(k))\right)}{\partial u(k)} + \left(\frac{\partial x(k+1)}{\partial u(k)}\right)^T \frac{\partial V_i(x(k+1))}{\partial x(k+1)} = 0. \quad (44)$$

Therefore, for $i = 0, 1, \ldots$, the corresponding control law $v_i(x)$ can be obtained by solving the above equation, i.e.,

$$v_i(x(k)) = \bar{U}\varphi\left(-\frac{1}{2}(\bar{U}R)^{-1} g^T(x(k)) \frac{\partial V_i(x(k+1))}{\partial x(k+1)}\right). \quad (45)$$

From (45), we find that the control law $v_i(x)$ at each step of iteration has to be computed by $\partial V_i(x(k+1))/\partial x(k+1)$, which is not an easy task. Furthermore, at each iteration step of value function $V_{i+1}(x(k))$ in (10), there exists an integral term $2\int_0^{v_i(x(k))} \boldsymbol{\varphi}^{-T}(\bar{U}^{-1}s)\bar{U}Rds$ to compute, which is a large computing burden. Therefore, in the following, we will present another method called iterative dual heuristic programming (DHP) algorithm to implement the iterative adaptive dynamic programming algorithm.

Define the costate function $\lambda(x) = \partial V(x)/\partial x$, which is similar to the framework in [30]. Here, we assume that the value function $V(x)$ is smooth so that $\lambda(x)$ exists. Then, the recurrent iteration between (8) and (10) can be implemented as follows.

First, we start with an initial costate function $\lambda_0(\cdot) = 0$. Then, for $i = 0, 1, \ldots$, by substituting $\lambda_i(x) = \partial V_i(x)/\partial x$ into (45), we can obtain the corresponding control law $v_i(x)$ as

$$v_i(x(k)) = \bar{U}\varphi\Big(-\frac{1}{2}(\bar{U}R)^{-1} g^T(x(k))\lambda_i(x(k+1))\Big). \quad (46)$$

For $\lambda_{i+1}(x(k)) = \partial V_{i+1}(x(k))/\partial x(k)$, according to (10), we can obtain

$$\begin{aligned}
&\lambda_{i+1}(x(k)) \\
&= \frac{\partial \left(x(k)^T Q x(k) + W\left(v_i(x(k))\right)\right)}{\partial x(k)} + \left(\frac{\partial v_i(x(k))}{\partial x(k)}\right)^T \\
&\quad \times \frac{\partial \left(x(k)^T Q x(k) + W(v_i(x(k)))\right)}{\partial v_i(x(k))} \\
&\quad + \left(\frac{\partial x(k+1)}{\partial x(k)}\right)^T \frac{\partial V_i(x(k+1))}{\partial x(k+1)} \\
&\quad + \left(\frac{\partial v_i(x(k))}{\partial x(k)}\right)^T \left(\frac{\partial x(k+1)}{\partial v_i(x(k))}\right)^T \frac{\partial V_i(x(k+1))}{\partial x(k+1)} \\
&= \frac{\partial \left(x(k)^T Q x(k) + W(v_i(x(k)))\right)}{\partial x(k)} + \left(\frac{\partial v_i(x(k))}{\partial x(k)}\right)^T \\
&\quad \times \left[\frac{\partial \left(x(k)^T Q x(k) + W(v_i(x(k)))\right)}{\partial v_i(x(k))} \right. \\
&\qquad \left. + \left(\frac{\partial x(k+1)}{\partial v_i(x(k))}\right)^T \times \frac{\partial V_i(x(k+1))}{\partial x(k+1)}\right] \\
&\quad + \left(\frac{\partial x(k+1)}{\partial x(k)}\right)^T \frac{\partial V_i(x(k+1))}{\partial x(k+1)}. \quad (47)
\end{aligned}$$

According to (44) and (45), we have

$$\frac{\partial \left(x(k)^T Q x(k) + W(v_i(x(k)))\right)}{\partial v_i(x(k))} + \left(\frac{\partial x(k+1)}{\partial v_i(x(k))}\right)^T \times \frac{\partial V_i(x(k+1))}{\partial x(k+1)} = 0. \quad (48)$$

Therefore, (47) can further be written as

$$\lambda_{i+1}(x(k)) = \frac{\partial \left(x(k)^T Q x(k) + W(v_i(x(k)))\right)}{\partial x(k)} + \left(\frac{\partial x(k+1)}{\partial x(k)}\right)^T \frac{\partial V_i(x(k+1))}{\partial x(k+1)} \quad (49)$$

i.e.,

$$\lambda_{i+1}(x(k)) = 2Qx(k) + \left(\frac{\partial x(k+1)}{\partial x(k)}\right)^T \lambda_i(x(k+1)). \quad (50)$$

Therefore, the iteration between (46) and (50) is an implementation of the iteration between (8) and (10). From (46), the control law v_i can directly be obtained by the costate function. Hence, the iteration of value function in (10) can be omitted in the implementation of this iterative algorithm. Considering the principle of DHP algorithm in [32], such iterative algorithm is called iterative DHP algorithm.

Next, we present a convergence analysis of the iteration between (46) and (50).

Theorem 4: Define the value function sequence $\{V_i\}$ by (10) with $V_0(\cdot) = 0$. Define the control law sequence $\{v_i\}$ as in (46) and update the costate function sequence $\{\lambda_i\}$ as in (50) with

$\lambda_0(\cdot) = 0$. Furthermore, define the optimal value λ^* as the limit of the costate function λ_i when value function V_i approaches the optimal value V^*. Then, the costate function sequence $\{\lambda_i\}$ and the control law sequence $\{v_i\}$ are convergent as $i \to \infty$.

Proof: According to Theorems 1–3, we have proved that $\lim_{i\to\infty} V_i(x(k)) = V^*(x(k))$ and $V^*(x(k))$ satisfies the corresponding HJB equation, i.e., $V^*(x(k)) = \inf_{u(k)}\{x(k)^T Q x(k) + W(u(k)) + V^*(x(k+1))\}$. Therefore, we conclude that the value function sequence $\{V_i\}$ converges to the optimal value function of the discrete-time HJB equation, i.e., $V_i \to V^*$ as $i \to \infty$. With $\lambda_i(x(k)) = \partial V_i(x(k))/\partial x(k)$, we conclude that the corresponding costate function sequence $\{\lambda_i\}$ is also convergent with $\lambda_i \to \lambda^*$ as $i \to \infty$. Since the costate function is convergent, we can conclude that the corresponding control law sequence $\{v_i\}$ converges to the optimal control law v^* as $i \to \infty$. ■

Remark 1: In the iterative DHP algorithm, via the costate sequence (50), the corresponding control law sequence can directly be obtained by (46), which does not require the computation of $\partial V_i(x(k+1))/\partial x(k+1)$. Furthermore, in (10), there is an integral term $2\int_0^{v_i(x(k))} \boldsymbol{\varphi}^{-T}(\bar{U}^{-1}s)\bar{U}Rds$ to compute at each iterative step, which is not an easy task. However, in (50), the integral term has been removed, which greatly reduces the computational burden. On the other hand, in order to compute the costate function by (50), the internal dynamics $f(x(k))$ and $g(x(k))$ of the system are needed. In the implementation part of the algorithm, a model network is constructed to approximate the nonlinear dynamics of the system, which avoids the requirement of knowing $f(x(k))$ and $g(x(k))$.

2) RBF Neural Network Implementation of the Iterative DHP Algorithm: In the iterative DHP algorithm, the optimal control is difficult to solve analytically. For example, in (46), the control at step k is a function of costate at step $k+1$. Closed-form explicit solution is difficult to solve if not impossible. Therefore, we need to use parametric structures, such as neural networks, to approximate the costate function and the corresponding control law in the iterative DHP algorithm. In this paper, we choose radial basis function (RBF) neural networks to approximate the nonlinear functions.

An RBF neural network (RBFNN) [31] consists of three layers (input, hidden, and output). Each input value is assigned to a node in the input layer and passed directly to the hidden layer without weights. Nodes at the hidden layer are called RBF units, determined by a vector called center and a scalar called width. The Gaussian density function is used as an activation function for the hidden neurons. Then, linear output weights connect the hidden and output layers. The overall input–output equation of the RBFNN is given by

$$y_i = b_i + \sum_{j=1}^{h} w_{ji}\phi_j(X) \tag{51}$$

where X is the input vector, $\phi_j(X) = \exp(-\|X - C_j\|^2/{\sigma_j}^2)$ is the activation function of the jth RBF unit in the hidden layer, $C_j \in \Re^n$ is the center of the jth RBF unit, h is the number of RBF units, b_i and w_{ji} are the bias term and the weight between hidden and output layers, and y_i is the ith output in the m-dimensional space. Once the optimal RBF centers are established over a wide range of operating points of the plant, the width of the ith center in the hidden layer is calculated by the following:

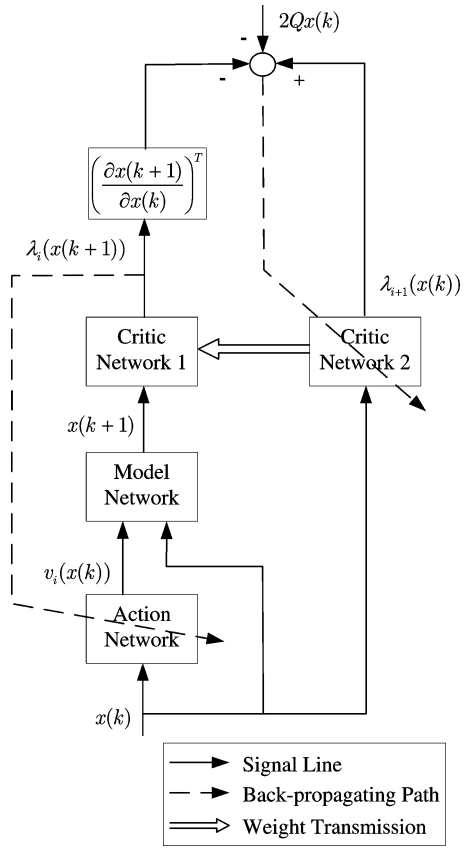


Fig. 1. The structure diagram of the iterative DHP algorithm.

$$\sigma_i = \sqrt{\frac{1}{h}\sum_{j=1}^{h}\sum_{k=1}^{n}(\|c_{ki} - c_{kj}\|)} \tag{52}$$

where c_{ki} and c_{kj} are the kth value of the center of the ith and jth RBF units, respectively. In (51) and (52), $\|\cdot\|$ represents the Euclidean norm. To avoid the extensive computational complexity during training, the batch mode k-means clustering algorithm is used to calculate the centers of the RBF units.

In order to implement the iterative adaptive dynamic programming algorithm, i.e., implement the iteration between (46) and (50), we employ RBF neural networks to approximate the costate function $\lambda_i(x)$ and the corresponding control law $v_i(x)$ at each iteration step i. In the implementation of the iterative DHP algorithm, there are three networks, which are model network, critic network and action network, respectively. All neural networks are chosen as RBF networks. The inputs of the model network are $x(k)$ and $v_i(x(k))$ and the inputs of the critic network and action network are $x(k+1)$ and $x(k)$, respectively. The diagram of the whole structure is shown in Fig. 1.

For unknown plants, before carrying out the iterative DHP algorithm, we first train a model network. For any given $x(k)$ and $\hat{v}_i(x(k))$, we can obtain $\hat{x}(k+1)$, and the output of the model network is denoted as

$$\hat{x}(k+1) = w_m^T \phi(I_m(k)) \tag{53}$$

where $I_m(k) = [x(k)^T \hat{v}_i(x(k))^T]^T$ is the input vector of the model network.

We define the error function of the model network as

$$e_m(k) = \hat{x}(k+1) - x(k+1). \tag{54}$$

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

$$E_m(k) = \frac{1}{2} e_m^T(k) e_m(k). \tag{55}$$

The weight updating rule for model network is chosen as a gradient-based adaptation rule

$$w_m(k+1) = w_m(k) - \alpha_m \left[\frac{\partial E_m(k)}{\partial w_m(k)}\right] \tag{56}$$

where α_m is the learning rate of the model network.

After the model network is trained, its weights are kept unchanged.

The critic network is used to approximate the costate function $\lambda_{i+1}(x)$. The output of the critic network is denoted as

$$\hat{\lambda}_{i+1}(x(k)) = w_{c(i+1)}^T \phi(x(k)). \tag{57}$$

The target costate function is given as in (50). Define the error function for the critic network as

$$e_{c(i+1)}(k) = \hat{\lambda}_{i+1}(x(k)) - \lambda_{i+1}(x(k)). \tag{58}$$

The objective function to be minimized for the critic network is

$$E_{c(i+1)}(k) = \frac{1}{2} e_{c(i+1)}^T(k) e_{c(i+1)}(k). \tag{59}$$

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

$$w_{c(i+1)}(j+1) = w_{c(i+1)}(j) - \alpha_c \left[\frac{\partial E_{c(i+1)}(k)}{\partial w_{c(i+1)}(j)}\right] \tag{60}$$

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

In the action network, the state $x(k)$ is used as input to create the control action as the output of the network. The output can be formulated as

$$\hat{v}_i(x(k)) = w_{ai}^T \phi(x(k)). \tag{61}$$

The target value of the control $v_i(x(k))$ is obtained by (46). So we can define the error function of the action network as

$$e_{ai}(k) = \hat{v}_i(x(k)) - v_i(x(k)). \tag{62}$$

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

$$E_{ai}(k) = \frac{1}{2} e_{ai}^T(k) e_{ai}(k). \tag{63}$$

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

$$w_{ai}(j+1) = w_{ai}(j) - \beta_a \left[\frac{\partial E_{ai}(k)}{\partial w_{ai}(j)}\right] \tag{64}$$

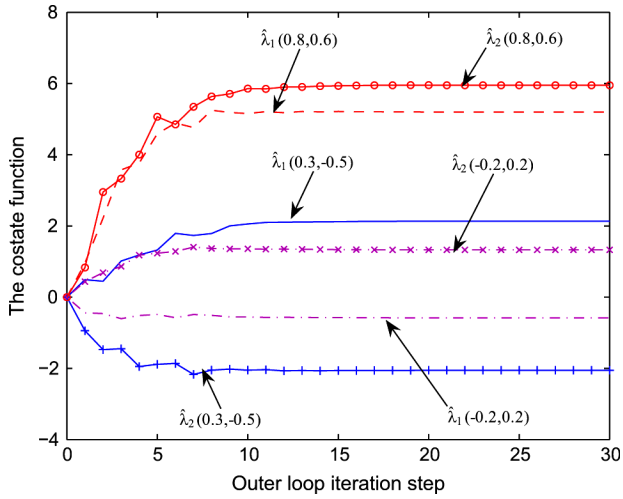


Fig. 2. The convergence process of the costate function at $x = (0.3, -0.5)$, $x = (-0.2, 0.2)$, and $x = (0.8, 0.6)$.

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

From the neural network implementation, we can find that in this iterative DHP algorithm, $\partial V_i(x(k+1))/\partial x(k+1)$ is replaced by $\hat{\lambda}_i(x(k+1))$, which is just the output of the critic network. Therefore, it is more accurate than computing by backpropagation through the critic network as in [3].

Remark 2: It should be noted that even though we use RBF networks for function approximation in our work, any function approximation schemes available in the literature will suffice. Although the RBF network seems to perform better than other networks in our simulation examples, it still has the scaling problem. The number of RBF units needed to cover the state space will grow exponentially with the dimensionality of the plant. Therefore, the difficulty of implementing the iterative algorithm will be more apparent when the dimensionality of the plant is high. However, this method is specially suitable for the domains where the state space is continuous and the controlled plants are Lipschitz continuous affine nonlinear systems, such as power systems (see the synchronous generators in [31]), chemical systems (see the continuously stirred tank reactor in [8]), and robot systems (see the two-link planar RR robot arm in [12]). Meanwhile, as far as we know, this is the first time that the optimal control problem under control constraint is addressed for discrete-time nonlinear systems in the framework of adaptive dynamic programming. The present near-optimal control law from the iterative algorithm developed in this paper is a constrained closed-loop feedback control law, which is preferred by most designers in practice.

3) Design Procedure of the Approximate Optimal Controller: Based on the iterative DHP algorithm, the design procedure of the optimal control scheme is summarized as follows.

1) Choose $i_{\max}$, $j_{\max}^a$, $j_{\max}^c$, ε_m, ε_0, $\bar{U}$, α_m, α_c, β_a and the weight matrices Q and R.
2) Construct the model network $\hat{x}(k+1) = w_m^T \phi(I_m(k))$ with the initial weight parameters w_{m0} chosen randomly in $[-0.1, 0.1]$ and train the model network with a random input vector uniformly distributed in the interval $[-1, 1]$

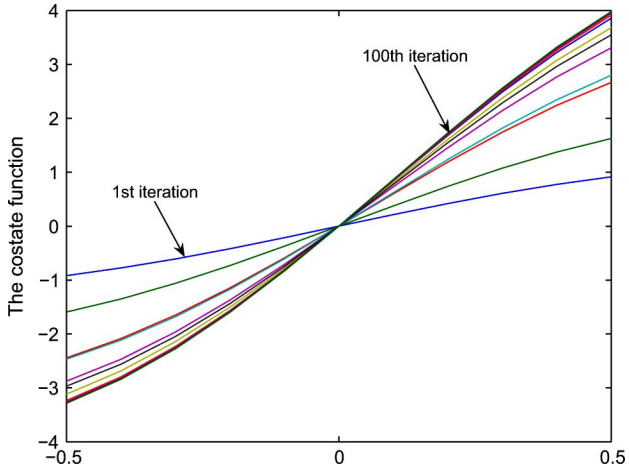


Fig. 3. The costate function $\hat{\lambda}_1(r,0)$ for $L=1,\ldots,100$, where $-0.5 \le r \le 0.5$.

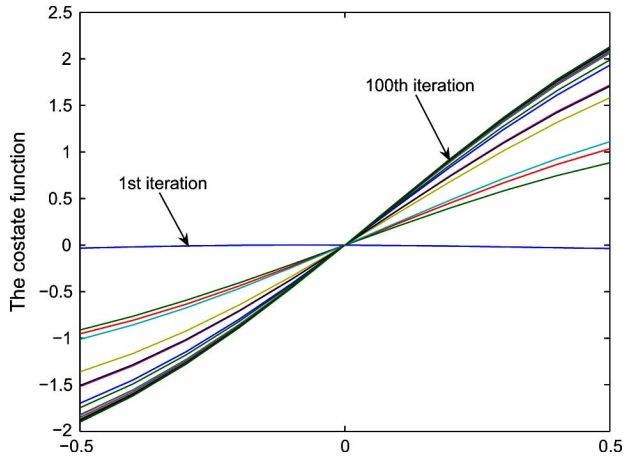


Fig. 4. The costate function $\hat{\lambda}_2(r,0)$ for $L=1,\ldots,100$, where $-0.5 \le r \le 0.5$.

and arbitrary initial state vector in $[-1,1]$ until the given accuracy ε_m is reached.

3) Set the iterative step $i=0$. Set the initial weight parameters of critic network w_{c0} as zero so that the initial value of the costate function $\lambda_0(\cdot)=0$, and initialize the action network with the weight parameters w_{a0} chosen randomly in $[-0.1,0.1]$.
4) Choose randomly an array of p state vector $\{x^{(1)}(k),x^{(2)}(k),\ldots,x^{(p)}(k)\}$ from the operation region and compute the corresponding output target $\{v_i(x^{(1)}(k)),v_i(x^{(2)}(k)),\ldots,v_i(x^{(p)}(k))\}$ by (46), where the state vector at the next time instant $\{(x^{(1)}(k+1),x^{(2)}(k+1),\ldots,x^{(p)}(k+1)\}$ is computed by the model network (53). With the same state vector $\{x^{(1)}(k),x^{(2)}(k),\ldots,x^{(p)}(k)\}$ and $\{x^{(1)}(k+1),x^{(2)}(k+1),\ldots,x^{(p)}(k+1)\}$, compute the resultant output target $\{\lambda_{i+1}(x^{(1)}(k)),\lambda_{i+1}(x^{(2)}(k)),\ldots,\lambda_{i+1}(x^{(p)}(k))\}$ by (50).
5) Set $w_{c(i+1)}=w_{ci}$. With the data set $(x^{(j)}(k),\lambda_{i+1}(x^{(j)}(k)))$, $j=1,2,\ldots,p$, update the weight parameters of the critic network $w_{c(i+1)}$ by (60) for $j^c_{\max}$ steps to get the approximate costate function $\hat{\lambda}_{i+1}$.
6) With the data set $(x^{(j)}(k),v_i(x^{(j)}(k)))$, $j=1,\ldots,p$, update the weight parameters of the action network w_{ai} by (64) for $j^a_{\max}$ steps to get the approximate control law $\hat{v}_i$.
7) If
$$\|\lambda_{i+1}(x^{(j)}(k))-\lambda_i(x^{(j)}(k))\|^2<\varepsilon_0, \qquad j=1,\ldots,p$$
go to step 9); otherwise, go to step 8).
8) If $i>i_{\max}$, go to step 9); otherwise, set $i=i+1$ and go to step 4).
9) Set the final approximate optimal control law $\hat{v}^*(x)=\hat{v}_i(x)$.
10) Stop.

As stated in Theorem 4, the iteration algorithm will be convergent with $\lambda_i(x)\to\lambda^*(x)$ and the control sequence $v_i(x)\to v^*(x)$ as $i\to\infty$. However, in practical applications, we cannot implement the iteration until $i\to\infty$. Actually, we iterate the algorithm for a max number $i_{\max}$ or with a prespecified accuracy ε_0 to test the convergence of the algorithm. In the above procedure, there are two levels of loops. The outer loop starts from step 3) and ends at step 8). There are two inner loops in steps 5) and 6), respectively. The inner loop of step 5) includes $j^c_{\max}$ iterative steps, and the inner loop of step 6) includes $j^a_{\max}$ iterative steps. The array of p state vector is chosen randomly at step 4). Suppose that the associated random probability density function is nonvanished everywhere. Then, we can assume that all the states will be explored. So we know that the resulting networks tend to satisfy (46) and (50) for all state vectors $x(k)$. The limits of $\hat{\lambda}_i$ and $\hat{v}_i$ will approximate the optimal costate vector λ^* and control vector v^*, respectively. The parameters ε_0 and $i_{\max}$ are chosen by the designer. The smaller the value of ε_0 is set, the more accurate the costate function and the optimal control law will be. If the condition set in step 7) is satisfied, it implies that the costate function sequence is convergent with the prespecified accuracy. The larger the value of $i_{\max}$ in step 8) is set, the more accurate the obtained control law $\hat{v}(x)$ will be at the price of increased computational burden.

IV. Simulation Studies

In this section, two examples are provided to demonstrate the effectiveness of the control scheme developed in this paper.

Example 1 (Nonlinear Discrete-Time System): Consider the following nonlinear system [12]:

$$x(k+1)=f(x(k))+g(x(k))u(k) \tag{65}$$

where

$$f(x(k))=\begin{bmatrix}-0.8x_2(k)\\ \sin(0.8x_1(k)-x_2(k))+1.8x_2(k)\end{bmatrix}$$
$$g(x(k))=\begin{bmatrix}0\\ -x_2(k)\end{bmatrix}$$

and assume that the control constraint is set to $|u|\le 0.3$.

Define the performance functional as

$$J(x(k),u(\cdot))=\sum_{i=k}^{\infty}\left\{x(i)^TQx(i)+2\int_0^{u(i)}\tanh^{-T}(\bar{U}^{-1}s)\bar{U}Rds\right\} \tag{66}$$

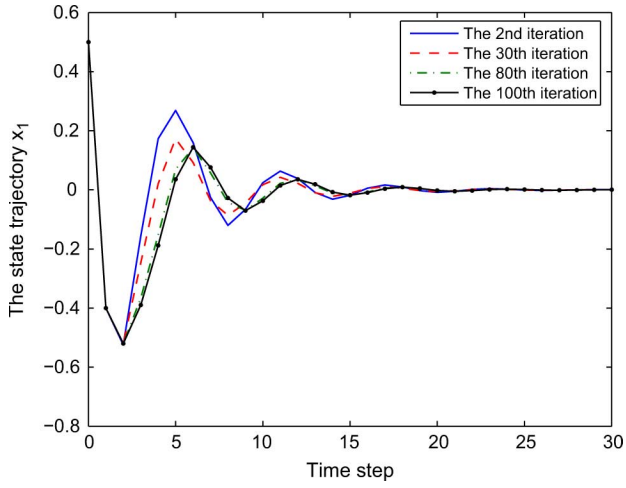


Fig. 5. The state trajectory x_1.

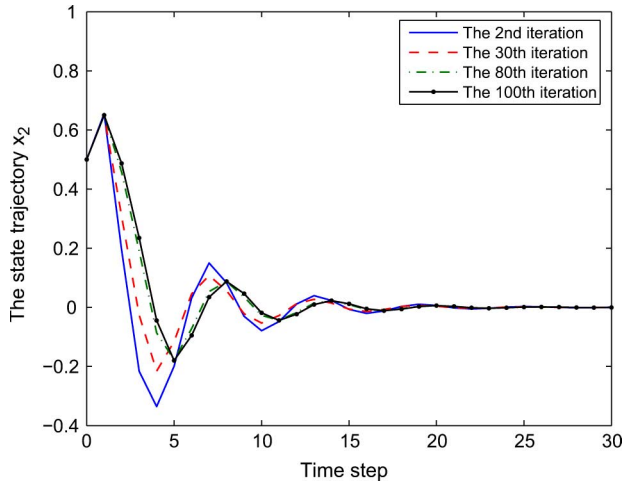


Fig. 6. The state trajectory x_2.

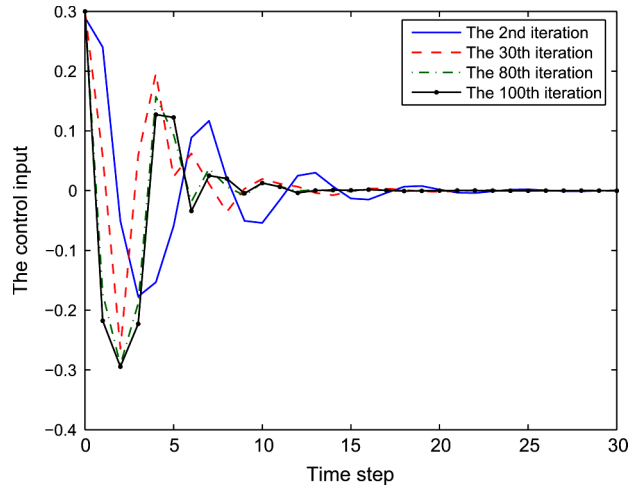


Fig. 7. The control input u.

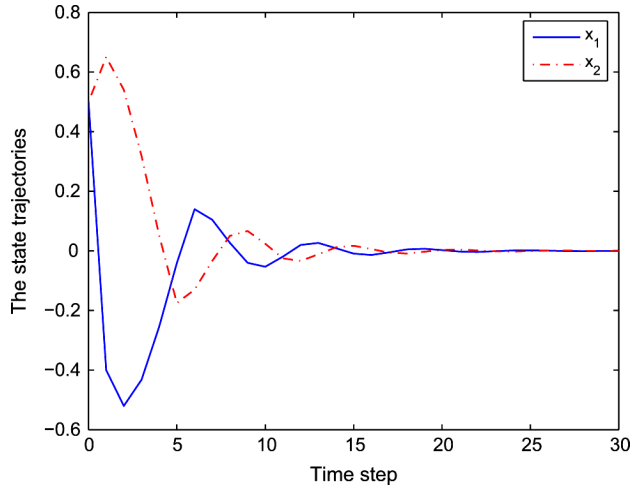


Fig. 8. The state variables curves without considering the actuator saturation in the controller design.

where $\bar{U} = 0.3$, and the weight matrices are chosen as

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \text{and} \quad R = 0.5.$$

First, we perform the simulation of iterative adaptive dynamic programming algorithm. In this iterative algorithm, we choose RBF neural networks as the critic network, the action network, and the model network with the structures of 2–9–2, 2–9–1, and 3–9–2, respectively. The training sets are selected as $-1 \leq x_1 \leq 1$ and $-1 \leq x_2 \leq 1$, which is the operation region of the system. It should be mentioned that the model network should be trained first. The initial state vectors are chosen randomly in $[-1, 1]$. Under the learning rate of $\alpha_m = 0.1$, the model network is trained until the given accuracy $\varepsilon_m = 10^{-6}$ is reached. After the training of the model network is completed, the weights are kept unchanged. Then, the critic network and the action network are trained with the learning rates $\alpha_c = \beta_a = 0.1$ and the inner-loop iteration number $j^c_{\max} = j^a_{\max} = 2000$. Meanwhile, the prespecified accuracy ε_0 is set to 10^{-20}. Denote the outer-loop iteration number as L. After implementing the outer-loop iteration for $L = i_{\max} = 100$, the convergence curves of the costate function are shown in Figs. 2–4. It can be seen that the costate function is basically convergent with the outer-loop iteration $L > 15$. In order to compare the different actions of the control laws obtained under different outer-loop iteration numbers, for the same initial state vector $x_1(0) = 0.5$ and $x_2(0) = 0.5$, we apply different control laws to the plant for 30 time steps and obtain the simulation results as follows. The state curves are shown in Figs. 5 and 6, and the corresponding control inputs are shown in Fig. 7. It can be seen that the system responses are improved when the outer-loop iteration number L is increased. When $L > 80$, the system responses only improve slightly in performance.

It should be mentioned that in order to show the convergence characteristics of the iterative process more clearly, we set the required accuracy ε_0 to a very small number 10^{-20} and we set the max iteration number to twice of what is needed. In this way, the given accuracy ε_0 did not take effect even when the max iteration number is reached. Therefore, it seems that the max iteration number $i_{\max}$ becomes the stopping criterion in this case. If the designer wants to save the running time, the prespecified accuracy ε_0 can be set to a normal value so that

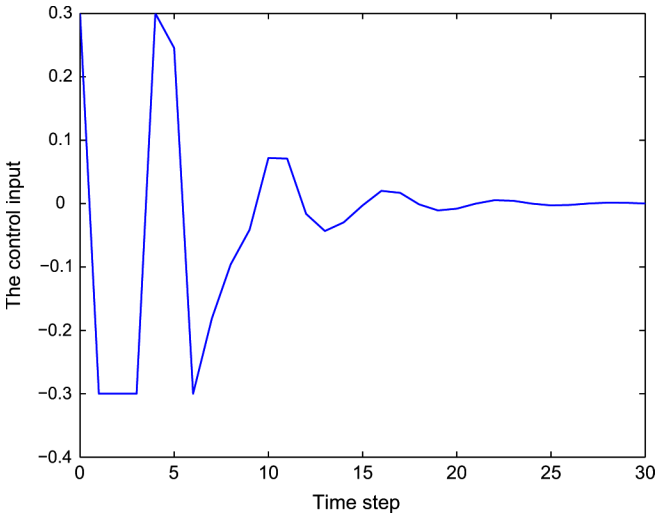


Fig. 9. The control input curve without considering the actuator saturation in the controller design.

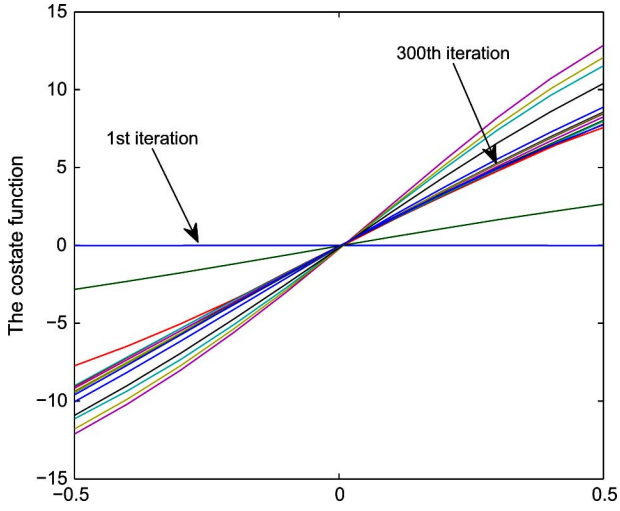


Fig. 11. The costate function $\hat{\lambda}_1(0,r)$ for $L = 1, 16, 31, \ldots, 286, 300$, where $-0.5 \leq r \leq 0.5$.

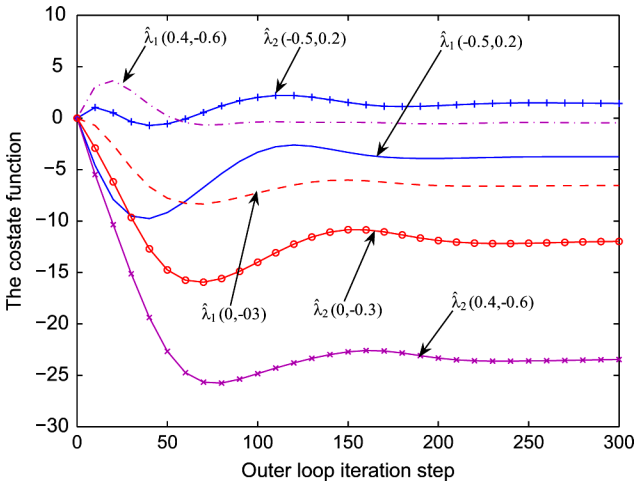


Fig. 10. The convergence process of the costate function at $x = (-0.5, 0.2)$, $x = (0.4, -0.6)$, and $x = (0, -0.3)$.

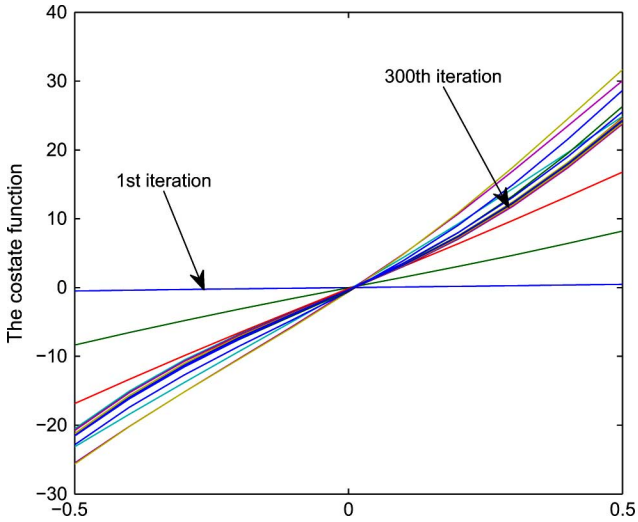


Fig. 12. The costate function $\hat{\lambda}_2(0,r)$ for $L = 1, 16, 31, \ldots, 286, 300$, where $-0.5 \leq r \leq 0.5$.

the iteration process will be stopped once the accuracy ε_0 is reached.

Moreover, in order to make comparison with the controller designed without considering the actuator saturation, we also present the system responses obtained by the controller designed regardless of the actuator saturation. The actuator saturation actually exists, and therefore, in the simulation, if the control input overruns the saturation bound, it is limited to the bounded value. After simulation, the state curves is shown in Fig. 8 and the control curve is shown in Fig. 9.

From the simulation results, we can see that the iterative costate function sequences do converge to the optimal ones with very fast speed, which also indicates the validity of the iterative ADP algorithm for dealing with constrained nonlinear systems. Comparing Fig. 7 with Fig. 9, we can see that in Fig. 7, the restriction of actuator saturation has been overcome successfully, but in Fig. 9, the control input has overrun the saturation bound, and therefore, is limited to the bounded value. From this point, we can conclude that the proposed iterative ADP algorithm is effective in dealing with the constrained optimal control problem.

Example 2 (Mass-Spring System): Consider the following discrete-time nonlinear mass-spring system:

$$\begin{cases} x_1(k+1) = 0.05x_2(k) + x_1(k) \\ x_2(k+1) = -0.0005x_1(k) - 0.0335x_1^3(k) \\ \qquad\qquad +0.05u(k) + x_2(k) \end{cases} \tag{67}$$

where $x(k)$ is the state vector and $u(k)$ is the control input.

Define the performance functional as

$$J(x(k), u(\cdot)) = \sum_{i=k}^{\infty} \left\{ x(i)^T Q x(i) + 2 \int_0^{u(i)} \tanh^{-T}(\bar{U}^{-1}s) \bar{U} R ds \right\} \tag{68}$$

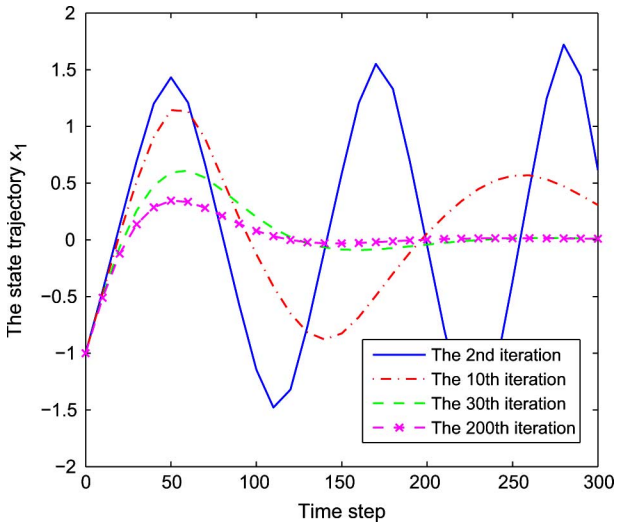


Fig. 13. The state trajectory x_1.

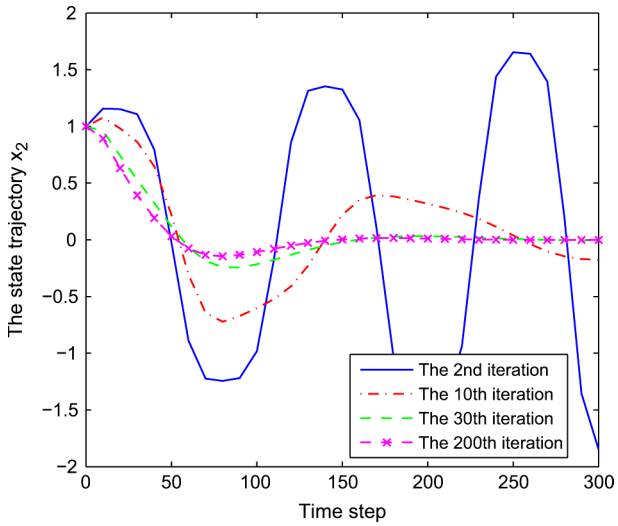


Fig. 14. The state trajectory x_2.

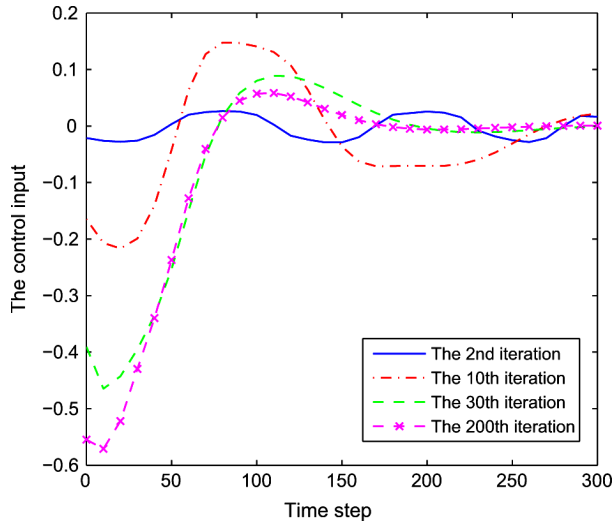


Fig. 15. The control input u.

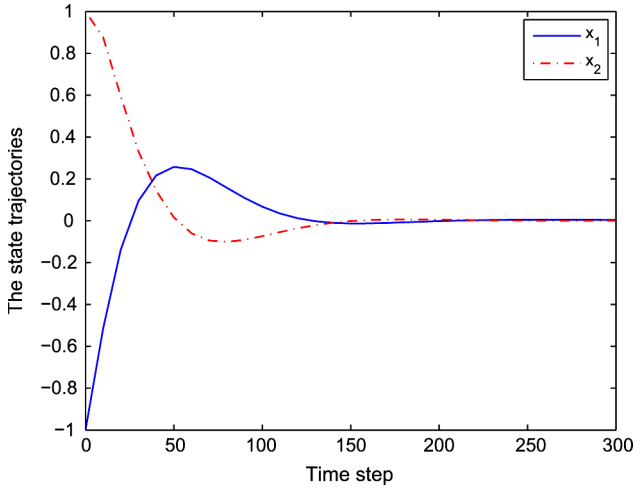


Fig. 16. The state curves without considering the actuator saturation in controller design.

where the control constraint is set to $\bar{U} = 0.6$, and the weight matrices are chosen as

$$Q = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix} \quad \text{and} \quad R = 1.$$

The training sets are $-1 \leq x_1 \leq 1$ and $-1 \leq x_2 \leq 1$. The critic network, the action network, and the model network are chosen as RBFNNs with the structures of 2–16–2, 2–16–1, and 3–16–2, respectively. In the training process, the learning rates are set to $\alpha_c = \beta_a = 0.1$. Other parameters are set the same as those in Example 1. After implementing the outer-loop iteration for $L = i_{\max} = 300$, the convergence curves of the costate function are shown in Figs. 10–12. It can be seen that the costate function is basically convergent with the outer-loop iteration $L > 200$. In order to compare the different actions of the control laws obtained under different outer-loop iteration numbers, for the same initial state vector $x_1(0) = -1$ and $x_2(0) = 1$, we apply different control laws to the plant for 300 time steps and obtain the simulation results as follows. The state curves are shown in Figs. 13 and 14, and the corresponding control inputs are shown in Fig. 15. It can be seen that the closed-loop system is divergent when using the control law obtained by $L = 2$, and the system's responses are improved when the outer-loop iteration number L is increased. When $L > 200$, the system responses basically remain unchanged with no significant improvement in performance.

In order to make comparison with the controller without considering the actuator saturation. We also present the controller designed by iterative ADP algorithm regardless of the actuator saturation. The state curves are shown in Fig. 16 and the control curve is shown in Fig. 17.

From the simulation results, we can see that the iterative costate function sequence does converge to the optimal one very fast. Comparing Fig. 15 with Fig. 17, we can find that in Fig. 15, the restriction of actuator saturation has been overcome successfully, which further verifies the effectiveness of the proposed iterative ADP algorithm.

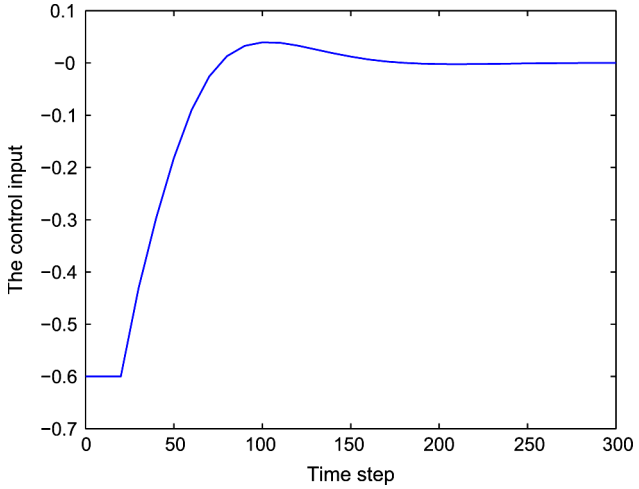


Fig. 17. The control curve without considering the actuator saturation in controller design.

V. Conclusion

In this paper, we developed an effective algorithm for finding the approximate optimal controller for a class of discrete-time constrained systems. First, a new type of nonquadratic functional was defined to deal with the control constraint, and then, the iterative adaptive dynamic programming algorithm was developed to seek for the costate function of the constrained optimal control problem with convergence analysis. Three neural networks were used as parametric structures to facilitate the implementation of the iterative algorithm. Simulation studies have demonstrated the effectiveness of the proposed optimal control 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, M. Abu-Khalaf, and F. L. Lewis, "Adaptive critic designs for discrete-time zero-sum games with application to H_∞ control," *IEEE Trans. Syst. Man Cybern. B, Cybern.*, vol. 37, no. 1, pp. 240–247, Feb. 2007.

[3] A. Al-Tamimi and F. L. Lewis, "Discrete-time nonlinear HJB solution using approximate dynamic programming: Convergence proof," in *Proc. IEEE Int. Symp. Approx. Dyn. Programm. Reinforcement Learn.*, Honolulu, HI, Apr. 2007, pp. 38–43.

[4] 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.

[5] J. Bagnell, S. Kakade, A. Ng, and J. Schneider, "Policy search by dynamic programming," in *Proc. 17th Annu. Conf. Neural Inf. Process. Syst.*, Vancouver, BC, Canada, Dec. 2003, vol. 16, pp. 831–838.

[6] M. Bardi and I. Capuzzo-Dolcetta, *Optimal Control and Viscosity Solutions of Hamilton-Jacobi-Bellman Equations*. Boston, MA: Birkhauser, 1997.

[7] A. G. Barto, R. S. Sutton, and C. W. Anderson, "Neuronlike adaptive elements that can solve difficult learning control problems," *IEEE Trans. Syst. Man Cybern.*, vol. SMC-13, no. 5, pp. 835–846, Sep./Oct. 1983.

[8] R. Beard, "Improving the closed-loop performance of nonlinear systems," Ph.D. dissertation, Electr. Eng. Dept., Rensselaer Polytech. Inst., Troy, NY, 1995.

[9] R. E. Bellman, *Dynamic Programming*. Princeton, NJ: Princeton Univ. Press, 1957.

[10] D. S. Bernstein, "Optimal nonlinear, but continuous, feedback control of systems with saturating actuators," *Int. J. Control*, vol. 62, no. 5, pp. 1209–1216, 1995.

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

[12] Z. Chen and S. Jagannathan, "Generalized Hamilton-Jacobi-Bellman formulation-based neural network control of affine nonlinear discrete-time systems," *IEEE Trans. Neural Netw*, vol. 19, no. 1, pp. 90–106, Jan. 2008.

[13] T. Cheng, F. L. Lewis, and M. Abu-Khalaf, "Fixed-final-time-constrained optimal control of nonlinear systems using neural network HJB approach," *IEEE Trans. Neural Netw*, vol. 18, no. 6, pp. 1725–1736, Nov. 2007.

[14] J. Dankert, Y. Lei, and J. Si, "A performance gradient perspective on approximate dynamic programming and its application to partially observable Markov decision processes," in *Proc. Int. Symp. Intell. Control*, Munich, Germany, Oct. 2006, pp. 458–463.

[15] 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, Jul. 2003.

[16] S. Ferrari and R. F. Stengel, "Online adaptive critic flight control," *J. Guid. Control Dyn.*, vol. 27, no. 5, pp. 777–786, 2004.

[17] T. Hanselmann, L. Noakes, and A. Zaknich, "Continuous-time adaptive critics," *IEEE Trans. Neural Netw.*, vol. 18, no. 3, pp. 631–647, May 2007.

[18] G. G. Lendaris and C. Paintz, "Training strategies for critic and action neural networks in dual heuristic programming method," in *Proc. Int. Joint Conf. Neural Netw.*, Houston, TX, Jun. 1997, vol. 2, pp. 712–717.

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

[20] B. Li and J. Si, "Robust dynamic programming for discounted infinite-horizon Markov decision processes with uncertain stationary transition matrices," in *Proc. IEEE Int. Symp. Approx. Dyn. Programm. Reinforcement Learn.*, Honolulu, HI, Apr. 2007, pp. 96–102.

[21] X. Liu and S. N. Balakrishnan, "Convergence analysis of adaptive critic based optimal control," in *Proc. Amer. Control Conf.*, Chicago, IL, Jun. 2000, pp. 1929–1933.

[22] 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.

[23] D. Liu, X. Xiong, and Y. Zhang, "Action-dependent adaptive critic designs," in *Proc. Int. Joint Conf. Neural Netw.*, Washington, DC, Jul. 2001, vol. 2, pp. 990–995.

[24] D. Liu and H. Zhang, "A neural dynamic programming approach for learning control of failure avoidance problems," *Int. J. Intell. Control Syst.*, vol. 10, no. 1, pp. 21–32, 2005.

[25] 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.

[26] S. E. Lyshevski, "Optimization of dynamic systems using novel performance functionals," in *Proc. 41st Conf. Decision Control*, Las Vegas, NV, Dec. 2002, pp. 753–758.

[27] S. E. Lyshevski, "Optimal control of nonlinear continuous-time systems: Design of bounded controllers via generalized nonquadratic functionals," in *Proc. Amer. Control Conf.*, Philadelphia, PA, Jun. 1998, pp. 205–209.

[28] S. E. Lyshevski, "Nonlinear discrete-time systems: Constrained optimization and application of nonquadratic costs," in *Proc. Amer. Control Conf.*, Philadelphia, PA, Jun. 1998, pp. 3699–3703.

[29] 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.

[30] R. Padhi, N. Unnikrishnan, X. Wang, and S. N. Balakrishnan, "A single network adaptive critic (SNAC) architecture for optimal control synthesis for a class of nonlinear systems," *Neural Netw.*, vol. 19, no. 10, pp. 1648–1660, Dec. 2006.

[31] J.-W. Park, R. G. Harley, and G. K. Venayagamoorthy, "Adaptive-critic-based optimal neurocontrol for synchronous generators in a power system using MLP/RBF neural networks," *IEEE Trans. Ind. Appl.*, vol. 39, no. 5, pp. 1529–1540, Sep./Oct. 2003.

[32] D. V. Prokhorov and D. C. Wunsch, "Adaptive critic designs," *IEEE Trans. Neural Netw.*, vol. 8, no. 5, pp. 997–1007, Sep. 1997.

[33] A. Saberi, Z. Lin, and A. Teel, "Control of linear systems with saturating actuators," *IEEE Trans. Autom. Control*, vol. 41, no. 3, pp. 368–378, Mar. 1996.

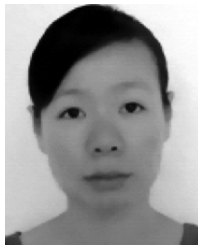
[34] G. Saridis and C. S. Lee, "An approximation theory of optimal control for trainable manipulators," *IEEE Trans. Syst. Man Cybern.*, vol. SMC-9, no. 2, pp. 152–159, Mar. 1979.

[35] G. N. Saridis and F. Y. Wang, "Suboptimal control of nonlinear stochastic systems," *Control-Theory Adv. Technol.*, vol. 10, no. 4, pp. 847–871, 1994.
[36] S. Shervais, T. T. Shannon, and G. G. Lendaris, "Intelligent supply chain management using adaptive critic learning," *IEEE Trans. Syst. Man Cybern. A, Syst. Humans*, vol. 33, no. 2, pp. 235–244, Mar. 2003.
[37] 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.
[38] H. Sussmann, E. D. Sontag, and Y. Yang, "A general result on the stabilization of linear systems using bounded controls," *IEEE Trans. Autom. Control*, vol. 39, no. 12, pp. 2411–2425, Dec. 1994.
[39] F.-Y. Wang and G. N. Saridis, "Suboptimal control for nonlinear stochastic systems," in *Proc. 31st IEEE Conf. Decision Control*, Tucson, AZ, Dec. 1992, pp. 1856–1861.
[40] C. Watkins, "Learning from delayed rewards," Ph.D. dissertation, Dept. Psychol., Cambridge University, Cambridge, U.K., 1989.
[41] 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.
[42] 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.
[43] P. J. Werbos, "Using ADP to understand and replicate brain intelligence: The next level design," in *Proc. IEEE Int. Symp. Approx. Dyn. Programm. Reinforcement Learn.*, Honolulu, HI, Apr. 2007, pp. 209–216.
[44] B. Widrow, N. Gupta, and S. Maitra, "Punish/reward: Learning with a critic in adaptive threshold systems," *IEEE Trans. Syst. Man Cybern.*, vol. SMC-3, no. 5, pp. 455–465, Sep. 1973.
[45] V. Yadav, R. Padhi, and S. N. Balakrishnan, "Robust/optimal temperature profile control of a high-speed aerospace vehicle using neural networks," *IEEE Trans. Neural Netw.*, vol. 18, no. 4, pp. 1115–1128, Jul. 2007.
[46] Q. Yang and S. Jagannathan, "Online reinforcement learning neural network controller design for nanomanipulation," in *Proc. IEEE Symp. Approx. Dyn. Programm. Reinforcement Learn.*, Honolulu, HI, Apr. 2007, pp. 225–232.
[47] H. Zhang, Q. Wei, and Y. Luo, "A novel infinite-time optimal tracking control scheme for a class of discrete-time nonlinear system based on greedy HDP iteration algorithm," *IEEE Trans. Syst. Man Cybern. B, Cybern.*, vol. 38, no. 4, pp. 937–942, Aug. 2008.

Huaguag Zhang (SM'04) received the B.S. and M.S. degrees in control engineering from Northeastern Dianli University of China, Jilin City, China, in 1982 and 1985, respectively, and the Ph.D. degree in thermal power engineering and automation from Southeast University, Nanjing, China, in 1991.

He joined the Department of Automatic Control, Northeastern University, Shenyang, China, in 1992, as a Postdoctoral Fellow for two years. Since 1994, he has been a Professor and Head of the Institute of Electric Automation, School of Information Science and Engineering, Northeastern University, Shenyang, China. He has authored and coauthored over 200 journal and conference papers and four monographs and coinvented 20 patents. His main research interests include fuzzy control, stochastic system control, neural-networks-based control, nonlinear control, and their applications.

Dr. Zhang is an Associate Editor of *Automatica*, *Neurocomputing*, the IEEE TRANSACTIONS ON FUZZY SYSTEMS, and the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART B: CYBERNETICS. He was awarded the Outstanding Youth Science Foundation Award from the National Natural Science Foundation of China in 2003. He was named the Cheung Kong Scholar by the Education Ministry of China in 2005.



Yanhong Luo (M'09) received the B.S. degree in automation control, the M.S. degree and the Ph.D. degree in control theory and control engineering from Northeastern University, Shenyang, China, in 2003, 2006, and 2009, respectively.

She is currently working in Northeastern University as an Associate Professor. Her research interests include fuzzy control, neural networks adaptive control, approximate dynamic programming, and their industrial application.

Derong Liu (S'91–M'94–SM'96–F'05) received the B.S. degree in mechanical engineering from the 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 Institute of Automation, 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, Beijing, China, from 1987 to 1990. He was a Staff Fellow with General Motors Research and Development Center, Warren, MI, from 1993 to 1995. He was an Assistant Professor in the Department of Electrical and Computer Engineering, Stevens Institute of Technology, Hoboken, NJ, from 1995 to 1999. He joined the University of Illinois at Chicago in 1999, where he became a Full Professor of Electrical and Computer Engineering and of Computer Science in 2006. He has published nine books (five research monographs and four edited volumes).

Dr. Liu was selected into the "100 Talents Program" by the Chinese Academy of Sciences in 2008. He is an Associate Editor of *Automatica*. He was General Chair for the 2007 International Symposium on Neural Networks (Nanjing, China). He was a member of the Conference Editorial Board of the IEEE Control Systems Society (1995–2000), an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—PART I: FUNDAMENTAL THEORY AND APPLICATIONS (1997–1999), the IEEE TRANSACTIONS ON SIGNAL PROCESSING (2001–2003), and the IEEE COMPUTATIONAL INTELLIGENCE MAGAZINE (2006–2009), and the Letters Editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS (2004–2008). He was the Editor of the IEEE Computational Intelligence Society's *Electronic Letter* (2004–2009). Currently, he is an Associate Editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS and the IEEE CIRCUITS AND SYSTEMS MAGAZINE. He is General Chair for 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 was an elected AdCom member of the IEEE Computational Intelligence Society (2006–2008). He received 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 University of Illinois (2006), and the Overseas Outstanding Young Scholar Award from the National Natural Science Foundation of China (2008).