Published in IET Control Theory and Applications
Received on 21st December 2011
Revised on 3rd September 2012
doi: 10.1049/iet-cta.2011.0783

ISSN 1751-8644

Optimal control for discrete-time affine non-linear systems using general value iteration

H. Li D. Liu

State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, People's Republic of China
E-mail: derong.liu@ia.ac.cn

Abstract: In this study, the authors propose a novel adaptive dynamic programming scheme based on general value iteration (VI) to obtain near optimal control for discrete-time affine non-linear systems with continuous state and control spaces. First, the selection of initial value function is different from the traditional VI, and a new method is introduced to demonstrate the convergence property and convergence speed of value function. Then, the control law obtained at each iteration can stabilise the system under some conditions. At last, an error-bound-based condition is derived considering the approximation errors of neural networks, and then the error between the optimal and approximated value functions can also be estimated. To facilitate the implementation of the iterative scheme, three neural networks with Levenberg–Marquardt training algorithm are used to approximate the unknown system, the value function and the control law. Two simulation examples are presented to demonstrate the effectiveness of the proposed scheme.

1 Introduction

Feedback and optimisation are two of the fundamental philosophies in control theory. It is important and appealing to design a control system that can satisfy the desired and optimal performance. Dynamic programming [1, 2] has been a powerful technique in solving optimal control problems by the famous principle of optimality for many years. However, the computation costs of dynamic programming for complex problems are often intense, that is, the well-known 'curse of dimensionality' [1]. Moreover, applications of dynamic programming in real-time online control are precluded as a result of the backward process of searching.

For non-linear systems, the optimal state feedback control law can be found by solving the Hamilton–Jacobi–Bellman (HJB) [3] equation, which reduces to Riccati equation for linear quadratic regulator problem. However, the theoretical solution of the HJB equation is difficult to obtain because of its inherently non-linear nature. Many efforts have been made to solve the HJB equation, such as viscosity solution theory [4] and Galerkin spectral approximation method [5]. Murray *et al.* [6] solved the HJB equation by employing a cost functional with an initial stabilising controller and exploring the entire state space at each iteration. Abu-Khalaf and Lewis [7] solved the optimal control for non-linear continuous-time systems with saturating actuators, where the value function was obtained by solving a sequence of cost functions satisfying Lyapunov equations (also called generalised HJB equation). Cheng *et al.* [8] proposed fixed-final time optimal control of non-linear systems by solving a time-varying HJB equation offline. Chen *et al.* [9] applied the successive approximation using generalised HJB for non-linear discrete-time systems. In all these cases, the control law at each iteration can be solved directly; so only one neural network is needed for approximating the generalised HJB solution.

As an effective intelligent scheme for solving the optimal control problems, adaptive dynamic programming (ADP) has received much attention during the last decades. ADP was first proposed by Werbos [10] as a method to solve optimal control problems forward-in-time. Excellent overviews of ADP are given in [11–13]. Existing ADP approaches can be classified into several main schemes [14]: heuristic dynamic programming (HDP), dual heuristic dynamic programming (DHP), globalised dual heuristic dynamic programming (GDHP) and their action-dependent versions such as ADHDP, ADDHP and ADGDHP. In [15], a cost-function-based single network adaptive critic architecture was presented.

Reinforcement learning (RL) [16] is a machine-learning method for an agent or controller to modify its actions based on the observed responses from the environment or system. In recent years, RL has been applied to feedback control [17–23]. On the other hand, iterative learning control can be seen in [24, 25]. The main algorithms of RL, that is, policy iteration (PI) and value iteration (VI) have been developed to solve the HJB equation of optimal control problems.

PI algorithm contains policy evaluation and policy improvement [26]. An initial stabilising control law is required, which is often difficult to obtain. In most applications, PI would require fewer iterations as a Netwon's method, but every iteration is more computationally demanding. Vrabie [27], Vrabie and Lewis [28] developed an online PI technique to obtain adaptive optimal control

for continuous-time systems without knowing the internal dynamics. In [29], the generalised PI for continuous-time systems was derived, which can include PI, VI and the optimistic PI algorithm. Vamvoudakis and Lewis [30] developed an online PI by simultaneous tuning of both action and critic neural networks (NNs), which is called synchronous PI. In [31], data-based optimal control was implemented online using novel PI and VI algorithms requiring only measured system outputs.

VI algorithm solves the optimal control problem without requirement of an initial stabilising control law. However, the stabilising control law cannot be obtained until the value function converges. Al-Tamimi *et al.* [32] derived a VI-based HDP algorithm to solve the optimal control problem for discrete-time non-linear systems with convergence proof. Dierks *et al.* [33] relaxed the need of partial knowledge of the system by online NN system identification and the convergence of action network was demonstrated considering the reconstruction error of NN. Zhang *et al.* [34] and Wang *et al.* [35] solved the optimal tracking and finite-horizon optimal control problem for discrete-time non-linear systems using greedy HDP algorithm. Zhang *et al.* [36] studied the near-optimal control for a class of discrete-time affine non-linear systems with control constraints by iterative DHP method.

ADP method uses a critic network for value function approximation and an action network for control law approximation. In [32], the NN with polynomial activation functions is used, but this often needs engineering experience and intuition. In [34], gradient descent algorithm is used to train BP NN. As the state space needs to be covered during training process and approximation error of NN is the key to ADP, we will use NN with Levenberg–Marquardt (LM) algorithm [37, 38] to implement the iterative process.

Leake and Liu [39] used an inequality version of HJB equation to derive bounds on the optimal cost. In [40, 41], Lincoln and Rantzer introduced a relaxed dynamic programming method to simplify computation based on upper and lower bounds of the optimal cost. It is known that there exists NN approximation error. However, the approximation error that influences the performance of VI is not considered except in [33]. So, we will take the approximation error of critic network into consideration in this paper.

In this paper, we propose a novel ADP scheme based on VI to obtain near-optimal control for discrete-time affine non-linear systems with continuous state and control space, which is called general VI here. First, the VI algorithm is given in a general framework, and the monotonicity property of value functions is demonstrated. The convergence property and convergence speed of value function are analysed. The iterative control law can also converge to the optimal control law, and stabilise the system under some conditions, which are easy to satisfy than PI. Second, an error-bound-based iteration condition is derived considering the influence of the NN approximation error, and the error between the optimal and approximated value functions can also be obtained. At last, to facilitate the implementation of the new VI algorithm, three NNs with LM tuning algorithm are used to approximate the unknown system, the value function and the control law. The HDP structure is used to implement the iteration.

The rest of the paper is organised as follows. Section 2 provides the problem formulation and discrete-time HJB equation for affine non-linear systems. In Section 3, we first derive the general VI algorithm, and then the convergence analysis is given. Section 4 discusses the NN implementation of the iterative ADP algorithm, where HDP is used. Section 5 presents two simulation examples to demonstrate the effectiveness of the proposed algorithm and is followed by concluding remarks in Section 6.

2 Problem formulation

Consider the discrete-time affine non-linear dynamical systems described by

$$x_{k+1} = f(x_k) + g(x_k)u(x_k), \quad k = 0, 1, 2, \ldots \tag{1}$$

where $x_k \in \mathbb{R}^n$ is the state vector and $u(x_k) = u_k \in \mathbb{R}^m$ is the control vector, $f(\cdot) \in \mathbb{R}^n$ and $g(\cdot) \in \mathbb{R}^{n\times m}$ are differentiable. We assume that the following assumptions hold throughout the paper.

Assumption 1: $f(0) = 0$, and the state feedback control law $u(x_k)$ satisfies $u(0) = 0$, that is, $x_k = 0$ is an equilibrium state of the system (1).

Assumption 2: $f + gu$ is Lipschitz continuous on a compact set $\Omega \subseteq \mathbb{R}^n$ containing the origin.

Assumption 3: System (1) is controllable in the sense that there exists a continuous control law on Ω that asymptotically stabilises the system.

In this paper, our goal is to find a state feedback control law $u(x_k)$ which can minimise the infinite horizon cost function as follows

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

where U is the positive-definite utility function, $U(0, 0) = 0$ and $U(x_k, u_k) \geq 0$, $\forall x_k$, u_k. Note that the control law $u(x_k)$ must not only stabilise the system on Ω but also guarantee that (2) is finite, that is, the control law must be admissible.

Definition 1 (admissible control law) [5]: A control law $u(x)$ is said to be admissible with respect to (2) on Ω if $u(x)$ is continuous on a compact set $\Omega \subseteq \mathbb{R}^n$, $u(0) = 0$, $u(x)$ stabilises (1) on Ω and for $\forall x_0 \in \Omega$, $J(x_0, u)$ is finite.

Let Ω_u be the set of all infinite horizon admissible control law associated with the controllable state set Ω. Then, we assume that the following Assumption 4 holds throughout the paper.

Assumption 4: For system (1), there exists at least one admissible control law $u(x_k)$, $\forall x_k \in \Omega$, that is, $\Omega_u \neq \emptyset$.

Define the optimal cost function as

$$J^*(x_k) = \inf\{J(x_k, u) : u(x_k) \in \Omega_u\} \tag{3}$$

IET Control Theory Appl., 2012, Vol. 6, Iss. 18, pp. 2725–2736
doi: 10.1049/iet-cta.2011.0783

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

$$J^*(x_k) = \min_{u_k}\{U(x_k, u_k) + J^*(x_{k+1})\} \tag{4}$$

The optimal control law $u^*(x_k)$ should satisfy

$$u^*(x_k) = \arg\min_{u_k}\{U(x_k, u_k) + J^*(x_{k+1})\} \tag{5}$$

In general, the utility function can be chosen as the quadratic form as follows

$$U(x_k, u_k) = x_k^{\mathrm{T}} Q x_k + u_k^{\mathrm{T}} R u_k \tag{6}$$

where $Q \in \mathbb{R}^{n\times n}$ and $R \in \mathbb{R}^{m\times m}$ are the positive-definite matrices.

The optimal control $u^*(x_k)$ satisfies the first-order necessary condition; so we obtain

$$\begin{aligned} u^*(x_k) &= -\frac{1}{2}R^{-1}\left(\frac{\partial x_{k+1}}{\partial u_k}\right)^{\mathrm{T}} \frac{\partial J^*(x_{k+1})}{\partial x_{k+1}} \\ &= -\frac{1}{2}R^{-1}g^{\mathrm{T}}(x_k)\frac{\partial J^*(x_{k+1})}{\partial x_{k+1}} \end{aligned} \tag{7}$$

Equation (4) reduces to Riccati equtation in the linear quadratic regulator problem. However, in the non-linear case, the cost function of the optimal control problem cannot be obtained. Therefore we will solve the discrete-time HJB equtaion by the general VI algorithm.

3 Optimal control scheme based on the general VI algorithm

This section consists of three subsections. First, the general VI algorithm is derived. Then, the corresponding convergence proof is presented. At last, the convergence analysis considering approximation error of NN is given.

3.1 Derivation of the general VI algorithm

Since direct solution of the HJB equation is computationally intensive, we present an iterative ADP algorithm in a general framework based on Bellman's principle of optimality.

First, we start with the initial value function $V_0(x_k) = x_k^{\mathrm{T}} P_0 x_k$, where P_0 is a positive-definite matrix. Then, we solve the control law $v_0(x_k)$ as follows

$$\begin{aligned} v_0(x_k) &= \arg\min_{u_k}\{x_k^{\mathrm{T}} Q x_k + u_k^{\mathrm{T}} R u_k + V_0(x_{k+1})\} \\ &= \arg\min_{u_k}\{x_k^{\mathrm{T}} Q x_k + u_k^{\mathrm{T}} R u_k + V_0(f(x_k) + g(x_k)u_k)\} \end{aligned} \tag{8}$$

Once the control law $v_0(x_k)$ is determined, we update the value function as

$$\begin{aligned} V_1(x_k) &= \min_{u_k}\{x_k^{\mathrm{T}} Q x_k + u_k^{\mathrm{T}} R u_k + V_0(x_{k+1})\} \\ &= x_k^{\mathrm{T}} Q x_k + v_0^{\mathrm{T}}(x_k) R v_0(x_k) + V_0(f(x_k) \\ &\quad + g(x_k)v_0(x_k)) \end{aligned} \tag{9}$$

Therefore for $i = 1, 2, \ldots$, the VI-based ADP algorithm iterates between a sequence of control laws $v_i(x_k)$

$$\begin{aligned} v_i(x_k) &= \arg\min_{u_k}\left\{x_k^{\mathrm{T}} Q x_k + u_k^{\mathrm{T}} R u_k + V_i(x_{k+1})\right\} \\ &= \arg\min_{u_k}\left\{x_k^{\mathrm{T}} Q x_k + u_k^{\mathrm{T}} R u_k + V_i(f(x_k) + g(x_k)u_k)\right\} \end{aligned} \tag{10}$$

and value functions $V_{i+1}(x_k)$

$$\begin{aligned} V_{i+1}(x_k) &= \min_{u_k}\{x_k^{\mathrm{T}} Q x_k + u_k^{\mathrm{T}} R u_k + V_i(x_{k+1})\} \\ &= x_k^{\mathrm{T}} Q x_k + v_i^{\mathrm{T}}(x_k) R v_i(x_k) + V_i(f(x_k) \\ &\quad + g(x_k)v_i(x_k)) \end{aligned} \tag{11}$$

Note that i is the iteration index and k is the time index. As a VI algorithm, this iterative ADP algorithm does not require an initial stabilising controller. The value function and control law are updated until they converge to the optimal ones. Furthermore, it should satisfy that $V_i(0) = 0$, $v_i(0) = 0$, $\forall i \geq 0$.

It should be mentioned that the initial value function here is chosen as $V_0(x_k) = x_k^{\mathrm{T}} P_0 x_k$ instead of $V_0(\cdot) = 0$ as in most traditional VI. In the next section we will prove the convergence of the iteration between (10) and (11), that is, $V_i \to J^*$ and $v_i \to u^*$ as $i \to \infty$.

3.2 Convergence analysis of the general VI algorithm

Lemma 1: Let $\{\mu_i\}$ be an arbitrary sequence of control laws and let $\{\Lambda_i\}$ be obtained by

$$\begin{aligned} \Lambda_{i+1}(x_k) &= x_k^{\mathrm{T}} Q x_k + \mu_i^{\mathrm{T}}(x_k) R \mu_i(x_k) \\ &\quad + \Lambda_i(f(x_k) + g(x_k)\mu_i(x_k)) \end{aligned} \tag{12}$$

Let $\{v_i\}$ and $\{V_i\}$ be defined in (10) and (11). If $V_0(x_k) = \Lambda_0(x_k) = x_k^{\mathrm{T}} P_0 x_k$, then $V_i(x_k) \leq \Lambda_i(x_k)$, $\forall i$.

Proof: It can easily be proved by noting that V_{i+1} is the result of minimising the right-hand side of (11) with respect to the control input u_k, whereas Λ_{i+1} is a result of arbitrary control input. □

Theorem 1 (monotonicity property): Define the control law sequence $\{v_i\}$ as in (10) and the value function sequence $\{V_i\}$ as in (11) with $V_0(x_k) = x_k^{\mathrm{T}} P_0 x_k$. If $V_0(x_k) \geq V_1(x_k)$ holds for any x_k, the value function sequence $\{V_i\}$ is a monotonically non-increasing sequence, that is, $V_{i+1} \leq V_i$, $\forall i \geq 0$. If $V_0(x_k) \leq V_1(x_k)$ holds for any x_k, the value function sequence $\{V_i\}$ is a monotonically non-decreasing sequence, that is, $V_i \leq V_{i+1}$, $\forall i \geq 0$.

Proof: First, suppose that $V_0(x_k) \geq V_1(x_k)$ holds for any x_k. Define a new sequence $\{\Phi_i\}$, which is updated according to

$$\begin{cases} \Phi_1(x_k) & = x_k^{\mathrm{T}} Q x_k + v_0^{\mathrm{T}}(x_k) R v_0(x_k) + \Phi_0(f(x_k) \\ & \quad + g(x_k)v_0(x_k)) \\ \Phi_{i+1}(x_k) & = x_k^{\mathrm{T}} Q x_k + v_{i-1}^{\mathrm{T}}(x_k) R v_{i-1}(x_k) \\ & \quad + \Phi_i(f(x_k) + g(x_k)v_{i-1}(x_k)), \quad i \geq 1 \end{cases} \tag{13}$$

with $\Phi_0(\cdot) = V_0(\cdot) = x_k^{\mathrm{T}} P_0 x_k$.

Now we use the mathematical induction to demonstrate that

$$\Phi_{i+1}(x_k) \le V_i(x_k), \quad \forall i \ge 0 \tag{14}$$

Considering $\Phi_1(x_k) = V_1(x_k)$ and

$$V_0(x_k) - \Phi_1(x_k) = V_0(x_k) - V_1(x_k) \ge 0 \tag{15}$$

we have

$$\Phi_1(x_k) \le V_0(x_k) \tag{16}$$

Then, we assume that it holds for $i-1$, that is, $\Phi_i(x_k) \le V_{i-1}(x_k)$, $\forall i \ge 1$ and $\forall x_k$. According to

$$\Phi_{i+1}(x_k) = x_k^{\mathrm{T}} Q x_k + v_{i-1}^{\mathrm{T}}(x_k) R v_{i-1}(x_k) + \Phi_i(x_{k+1}), \quad i \ge 1 \tag{17}$$

and

$$V_i(x_k) = x_k^{\mathrm{T}} Q x_k + v_{i-1}^{\mathrm{T}}(x_k) R v_{i-1}(x_k) + V_{i-1}(x_{k+1}), \quad i \ge 1 \tag{18}$$

we have

$$V_i(x_k) - \Phi_{i+1}(x_k) = V_{i-1}(x_{k+1}) - \Phi_i(x_{k+1}) \ge 0, \quad i \ge 1 \tag{19}$$

which implies

$$\Phi_{i+1}(x_k) \le V_i(x_k), \quad i \ge 1 \tag{20}$$

Considering $V_0(x_k) \ge \Phi_1(x_k)$, we have

$$\Phi_{i+1}(x_k) \le V_i(x_k), \quad i \ge 0 \tag{21}$$

According to Lemma 1, it is clear that

$$V_{i+1}(x_k) \le \Phi_{i+1}(x_k), \quad \forall i \ge 0 \tag{22}$$

Therefore

$$V_{i+1}(x_k) \le V_i(x_k), \quad \forall i, x_k \tag{23}$$

Thus, we complete the first part of the proof by mathematical induction.

Next, suppose that $V_0(x_k) \le V_1(x_k)$ holds for any x_k. Define a new sequence $\{\Gamma_i\}$, which is updated according to

$$\Gamma_{i+1}(x_k) = x_k^{\mathrm{T}} Q x_k + v_{i+1}^{\mathrm{T}}(x_k) R v_{i+1}(x_k) + \Gamma_i(x_{k+1}), \quad i \ge 0 \tag{24}$$

with $\Gamma_0(\cdot) = V_0(\cdot) = x_k^{\mathrm{T}} P_0 x_k$.

Similarly, we use the mathematical induction to demonstrate that

$$\Gamma_i(x_k) \le V_{i+1}(x_k), \quad \forall i \ge 0 \tag{25}$$

Considering

$$V_1(x_k) - \Gamma_0(x_k) = V_1(x_k) - V_0(x_k) \ge 0 \tag{26}$$

we have $\Gamma_0(x_k) \le V_1(x_k)$.

Then, we assume that it holds for $i-1$, that is, $\Gamma_{i-1}(x_k) \le V_i(x_k)$, $\forall i \ge 1$ and $\forall x_k$. According to

$$\Gamma_i(x_k) = x_k^{\mathrm{T}} Q x_k + v_i^{\mathrm{T}}(x_k) R v_i(x_k) + \Gamma_{i-1}(x_{k+1}), \quad i \ge 1 \tag{27}$$

and

$$V_{i+1}(x_k) = x_k^{\mathrm{T}} Q x_k + v_i^{\mathrm{T}}(x_k) R v_i(x_k) + V_i(x_{k+1}), \quad i \ge 1 \tag{28}$$

we have

$$V_{i+1}(x_k) - \Gamma_i(x_k) = V_i(x_{k+1}) - \Gamma_{i-1}(x_{k+1}) \ge 0, \quad i \ge 1 \tag{29}$$

which implies

$$\Gamma_i(x_k) \le V_{i+1}(x_k), \quad i \ge 1 \tag{30}$$

Considering $\Gamma_0(x_k) \le V_1(x_k)$, we have

$$\Gamma_i(x_k) \le V_{i+1}(x_k), \quad i \ge 0 \tag{31}$$

According to Lemma 1, it is easy to see that

$$V_i(x_k) \le \Gamma_i(x_k), \quad \forall i \ge 0 \tag{32}$$

Therefore

$$V_i(x_k) \le V_{i+1}(x_k), \quad \forall i \ge 0 \tag{33}$$

Thus, we complete the second part of the proof by mathematical induction. □

Remark 1: From Theorem 1, we can see that the monotonicity property of the value function V_i is determined by the relationship between V_0 and V_1, that is, $V_0 \ge V_1$ or $V_0 \le V_1$, $\forall x_k$. In the traditional VI algorithm, the initial value function is usually selected as $V_0(\cdot) = 0$. We can easily find that this is just a special case of our general scheme, that is, $V_0 \le V_1$. From this point, we know that it is easier to obtain a non-decreasing value function sequence. Besides, the monotonicity property is still valid if we can find that $V_i \ge V_{i+1}$ or $V_i \le V_{i+1}$ for any x_k and some i.

In [40, 41], only the convergence of VI is given. Next, we will demonstrate the uniform convergence of value function, and demonstrate that the control sequence converges to the optimal control law by a corollary.

Theorem 2 (convergence property)*:* Suppose the condition $0 \le J^*(f(x) + g(x)u(x)) \le \theta U(x,u)$ holds uniformly for some $0 < \theta < \infty$ and that $0 \le \alpha J^* \le V_0 \le \beta J^*$, $0 \le \alpha \le 1$ and $1 \le \beta < \infty$. The control law sequence $\{v_i\}$ and value function sequence $\{V_i\}$ are iteratively updated by (10) and (11). Then the value function V_i approaches J^* according to the inequalities

$$\left[1 + \frac{\alpha - 1}{(1+\theta^{-1})^i}\right] J^*(x) \le V_i(x) \le \left[1 + \frac{\beta - 1}{(1+\theta^{-1})^i}\right] J^*(x) \tag{34}$$

Moreover, the value function V_i converges to J^* uniformly on Ω.

Proof: First, we demonstrate that the system defined in this paper satisfies the conditions of Theorem 2. According to

doi: 10.1049/iet-cta.2011.0783

Assumption 2, the system state cannot jump to infinity by any one step of finite control input, that is, $f(x)+g(x)u(x)$ is finite. Considering Assumption 4 which shows that there exists an admissible control law that guarantees the cost function to be finite, we can derive that $J^*(f(x)+g(x)u(x))$ is finite for any finite state and control. As the $U(x,u)$ is a positive-definite function, there exists some $0<\theta<\infty$ that makes $0\le J^*(f(x)+g(x)u(x))\le \theta U(x,u)$ hold uniformly. For any finite initial value function V_0, there exist α and β such that $0\le \alpha J^*\le V_0\le \beta J^*$ is satisfied, where $0\le\alpha\le 1$ and $1\le\beta<\infty$. Next, we will demonstrate the left-hand side of the inequality (34) by mathematical induction, that is

$$\left[1+\frac{\alpha-1}{(1+\theta^{-1})^i}\right]J^*(x)\le V_i(x) \tag{35}$$

When $i=1$, since

$$\frac{\alpha-1}{1+\theta}(\theta U(x_k,u_k)-J^*(x_{k+1}))\le 0,\quad 0\le\alpha\le 1 \tag{36}$$

and $\alpha J^*\le V_0$, $\forall x_k$, we have

$$\begin{aligned}
V_1(x_k)&=\min_{u_k}\{U(x_k,u_k)+V_0(x_{k+1})\}\\
&\ge\min_{u_k}\{U(x_k,u_k)+\alpha J^*(x_{k+1})\}\\
&\ge\min_{u_k}\left\{\left(1+\theta\frac{\alpha-1}{1+\theta}\right)U(x_k,u_k)\right.\\
&\quad\left.+\left(\alpha-\frac{\alpha-1}{1+\theta}\right)J^*(x_{k+1})\right\}\\
&=\left[1+\frac{\alpha-1}{(1+\theta^{-1})}\right]\min_{u_k}\{U(x_k,u_k)+J^*(x_{k+1})\}\\
&=\left[1+\frac{\alpha-1}{(1+\theta^{-1})}\right]J^*(x_k)
\end{aligned} \tag{37}$$

Assume that the inequality (35) holds for $i-1$. Then, we have

$$\begin{aligned}
V_i(x_k)&=\min_{u_k}\{U(x_k,u_k)+V_{i-1}(x_{k+1})\}\\
&\ge\min_{u_k}\left\{U(x_k,u_k)+\left[1+\frac{\alpha-1}{(1+\theta^{-1})^{i-1}}\right]J^*(x_{k+1})\right\}\\
&\ge\min_{u_k}\left\{\left[1+\frac{(\alpha-1)\theta^i}{(\theta+1)^i}\right]U(x_k,u_k)\right.\\
&\quad\left.+\left[1+\frac{\alpha-1}{(1+\theta^{-1})^{i-1}}-\frac{(\alpha-1)\theta^{i-1}}{(\theta+1)^i}\right]J^*(x_{k+1})\right\}\\
&=\left[1+\frac{(\alpha-1)\theta^i}{(\theta+1)^i}\right]\min_{u_k}\left\{U(x_k,u_k)+J^*(x_{k+1})\right\}\\
&=\left[1+\frac{(\alpha-1)}{(1+\theta^{-1})^i}\right]J^*(x_k)
\end{aligned} \tag{38}$$

Thus, the left-hand side of the inequality (34) is proved and the right-hand side can be shown by the same procedure.

Lastly, we demonstrate the uniform convergence of value function as the iteration index i goes to ∞. When $i\to\infty$, for $0<\theta<\infty$, we have

$$\lim_{i\to\infty}\left[1+\frac{\alpha-1}{(1+\theta^{-1})^i}\right]J^*(x_k)=J^*(x_k) \tag{39}$$

and

$$\lim_{i\to\infty}\left[1+\frac{\beta-1}{(1+\theta^{-1})^i}\right]J^*(x_k)=J^*(x_k) \tag{40}$$

Define $V_\infty(x_k)=\lim_{i\to\infty}V_i(x_k)$. Then we can obtain

$$V_\infty(x_k)=J^*(x_k) \tag{41}$$

Hence, $V_i(x_k)$ converges pointwise to $J^*(x_k)$. As the Ω is compact, we can obtain the uniform convergence of value function immediately from Dini's theorem [42].

The proof is completed. □

Remark 2: From Theorem 2, we can find upper and lower bounds for every iterative value function based on the optimal cost function. As the iteration index i increases, the upper bound will exponentially approach the lower bound. When the iteration index i goes to ∞, the upper bound will be nearly equal to the lower bound, which is just the optimal cost. In addition, we can also find the convergence speed of the value function, which is not given in [32, 34, 36]. According to the inequality (34), smaller θ will lead to faster convergence speed of the value function. Moreover, it should be mentioned that conditions of Theorem 2 can be satisfied according to Assumptions 1–4, which are mild for general control problems.

Specially, when $0\le V_0(x_k)\le V_1(x_k)$, $\forall x_k$, according to Theorems 1 and 2, we can deduce that $V_0(x_k)\le J^*(x_k)$. Thus, the constants α and β satisfy $0<\alpha\le 1$ and $\beta=1$. Then, the corresponding inequality becomes

$$\left[1+\frac{\alpha-1}{(1+\theta^{-1})^i}\right]J^*(x)\le V_i(x)\le J^*(x) \tag{42}$$

Note that larger α will lead to faster convergence speed of the value function.

When $V_0(x_k)\ge V_1(x_k)$, $\forall x_k$, according to Theorems 1 and 2, we can deduce that $V_0(x_k)\ge J^*(x_k)$. So, the constants α and β satisfy $\alpha=1$ and $1\le\beta$. Then, the corresponding inequality becomes

$$J^*(x)\le V_i(x)\le\left[1+\frac{\beta-1}{(1+\theta^{-1})^i}\right]J^*(x) \tag{43}$$

Note that smaller β will lead to faster convergence speed of the value function.

According to the results of Theorem 2, we can derive the following corollary.

Corollary 1: Define the control sequence $\{v_i\}$ as in (10) and the value function $\{V_i\}$ as in (11) with $V_0(\cdot)=x_k^{\mathrm{T}}P_0x_k$. If the system state x_k is controllable, then the control sequence $\{v_i\}$ converges to the optimal control law u^* as $i\to\infty$, that is, $\lim_{i\to\infty}v_i(x_k)=u^*(x_k)$.

Proof: According to Theorem 2, we have proved that $\lim_{i\to\infty}V_i(x_k)=V_\infty(x_k)=J^*(x_k)$, so

$$V_\infty(x_k)=\min_{u_k}\{x_k^{\mathrm{T}}Qx_k+u_k^{\mathrm{T}}Ru_k+V_\infty(x_{k+1})\} \tag{44}$$

That is to say the value function sequence $\{V_i\}$ converges to the optimal value function of the discrete-time HJB equation.

Considering (5) and (10), the corresponding control law $\{v_i\}$ converges to the optimal control law u^* as $i \to \infty$.

In the following, we will complete the stability analysis for the non-linear system under the condition of control Lyapunov function. □

Theorem 3: The control law function sequence $\{v_i\}$ and value function sequence $\{V_i\}$ are iteratively updated by (10) and (11). If $V_0(x_k) = x_k^{\mathrm{T}} P_0 x_k \geq V_1(x_k)$ holds for any controllable x_k, then the value function $V_i(x_k)$ is a Lyapuonv function and the system using the control law $v_i(x_k)$ is asymptotically stable.

Proof: First, according to $V_0 \geq V_1$ and Theorem 1, we have $V_i(x_k) \geq V_{i+1}(x_k) \geq U((x_k), v_i(x_k))$, $\forall i$. As the $U(x_k, v_i(x_k))$ is a positive-definite function and $V_i(0) = 0$, $V_i(x_k)$ is also a positive-definite function.

Second, we have $V_i(x_{k+1}) - V_i(x_k) \leq V_i(x_{k+1}) - V_{i+1}(x_k) = -U(x_k, v_i(x_k)) \leq 0$. By the Lyapunov stability criteria [43], $V_i(x_k)$ is a Lyapunov function, and the system using the control law $v_i(x_k)$ is asymptotically stable.

$v_0(x_k)$ satisfies the first-order necessary condition, which is given by the gradient of the right-hand side of (8) with respect to u_k as

$$\frac{\partial \left(x_k^{\mathrm{T}} Q x_k + u_k^{\mathrm{T}} R u_k\right)}{\partial u_k} + \left(\frac{\partial x_{k+1}}{\partial u_k}\right)^{\mathrm{T}} \frac{\partial V_0(x_{k+1})}{\partial x_{k+1}} = 0 \tag{45}$$

that is

$$2Ru_k + 2g^{\mathrm{T}}(x_k) P_0 (f(x_k) + g(x_k) u_k) = 0 \tag{46}$$

Then, we can solve for $v_0(x_k)$ as

$$v_0(x_k) = -(g^{\mathrm{T}}(x_k) P_0 g(x_k) + R)^{-1} g^{\mathrm{T}}(x_k) P_0 f(x_k) \tag{47}$$

The control law $v_0(x_k)$ exists since P_0 and R are both positive-definite matrices. Combining with (9), we can easily obtain the condition that control Lyapunov function should satisfy. □

Remark 3: If the condition $V_0 \geq V_1$ holds, $V_0(x_k) = x_k^{\mathrm{T}} P_0 x_k$ is called control Lyapunov function, which means that the associated feedback control law $v_0(x_k)$ can make the closed-loop system stable. Compared with PI algorithms, this condition $V_0 \geq V_1$ is easier to satisfy than an initial stabilising control law. In particular, we can just make $P_0 = \gamma I_n$, $\gamma \geq 0$ and I_n is $n \times n$ identity matrix. By choosing a large γ, we can make $V_0 \geq V_1$ satisfied. Besides, similar to [44, 45], it should be mentioned that the condition $V_0 \geq V_1$ in Theorem 3 cannot be replaced by $V_0 \geq J^*$, because the non-increasing property of value function is guaranteed by $V_0 \geq V_1$. However, if the condition $V_0 \leq V_1$ holds, we cannot derive that $v_i(x_k)$ is a stable and admissible control for non-linear systems. For discrete linear time-invariant systems, Primbs and Nevistic [46] demonstrated that there exists a finite i^*, that the closed-loop system is asymptotically stable for all $i \geq i^*$.

3.3 Convergence analysis considering NN approximation errors

When the controlled system is linear and the cost function is quadratic, we can directly obtain a linear control law. In the non-linear case, however, directly obtaining a control law becomes impossible. Therefore we need to use function approximation structure, such as NN, to approximate both $v_i(x_k)$ and $V_i(x_k)$ at each iteration. However, there exist NN approximate errors. In the following, we will take the approximate errors of critic network into consideration based on Rantzer [41].

Let ε be a small positive number, $\underline{\varepsilon} = 1 - \varepsilon$ and $\overline{\varepsilon} = 1 + \varepsilon$. Let $\tilde{V}_i$ and $\tilde{v}_i$ stand for the NN approximation of V_i and v_i, respectively. We start with the initial cost function $\tilde{V}_0(x_k) = x_k^{\mathrm{T}} P_0 x_k$, P_0 is a positive-definite matrix. Then, we solve for the control law $\tilde{v}_0(x_k)$ as follows

$$\tilde{v}_0(x_k) = \arg\min_{u_k} \{U(x_k, u_k) + \tilde{V}_0(x_{k+1})\} \tag{48}$$

Once the control law $\tilde{v}_0(x_k)$ is determined, we make the value function $\tilde{V}_1(x_k)$ satisfy

$$\begin{aligned} \min_{u_k} \{\underline{\varepsilon} U(x_k, u_k) + \tilde{V}_0(x_{k+1})\} &\leq \tilde{V}_1(x_k) \\ &\leq \min_{u_k} \{\overline{\varepsilon} U(x_k, u_k) + \tilde{V}_0(x_{k+1})\} \end{aligned} \tag{49}$$

Therefore for $i = 1, 2, \ldots$, the algorithm iterates between the control law $\tilde{v}_i(x_k)$

$$\tilde{v}_i(x_k) = \arg\min_{u_k} \{U(x_k, u_k) + \tilde{V}_i(x_{k+1})\} \tag{50}$$

and the value function $\tilde{V}_{i+1}(x_k)$

$$\begin{aligned} \min_{u_k} \{\underline{\varepsilon} U(x_k, u_k) + \tilde{V}_i(x_{k+1})\} &\leq \tilde{V}_{i+1}(x_k) \\ &\leq \min_{u_k} \{\overline{\varepsilon} U(x_k, u_k) + \tilde{V}_i(x_{k+1})\} \end{aligned} \tag{51}$$

The existence of ε can be guaranteed, since the following inequality holds

$$\underline{\varepsilon} U(x_k, u_k) \leq U(x_k, u_k) \leq \overline{\varepsilon} U(x_k, u_k)$$

Define the target of $\tilde{V}_{i+1}(x_k)$ as $\tilde{V}^*_{i+1}(x_k)$, then

$$\tilde{V}^*_{i+1}(x_k) = U(x_k, \tilde{v}_i(x_k)) + \tilde{V}_i(x_{k+1}) \tag{52}$$

where $x_{k+1} = f(x_k) + g(x_k)\tilde{v}_i(x_k)$. Define

$$\tilde{v}_i^{\underline{\varepsilon}}(x_k) = \arg\min_{u_k} \left\{\underline{\varepsilon} U(x_k, u_k) + \tilde{V}_i(x_{k+1})\right\} \tag{53}$$

and

$$\tilde{V}^{\underline{\varepsilon}}_{i+1}(x_k) = \underline{\varepsilon} U(x_k, \tilde{v}_i^{\underline{\varepsilon}}(x_k)) + \tilde{V}_i(x_{k+1}) \tag{54}$$

where $x_{k+1} = f(x_k) + g(x_k)\tilde{v}_i^{\underline{\varepsilon}}(x_k)$. Define

$$\tilde{v}_i^{\overline{\varepsilon}}(x_k) = \arg\min_{u_k} \{\overline{\varepsilon} U(x_k, u_k) + \tilde{V}_i(x_{k+1})\} \tag{55}$$

and

$$\tilde{V}^{\overline{\varepsilon}}_{i+1}(x_k) = \overline{\varepsilon} U(x_k, \tilde{v}_i^{\overline{\varepsilon}}(x_k)) + \tilde{V}_i(x_{k+1}) \tag{56}$$

where $x_{k+1} = f(x_k) + g(x_k)\tilde{v}_i^{\overline{\varepsilon}}(x_k)$.

doi: 10.1049/iet-cta.2011.0783

Since $\underline{\varepsilon} < 1$ and $\overline{\varepsilon} > 1$, we have

$$\tilde{V}_{i+1}^{\underline{\varepsilon}}(x_k) \le \tilde{V}_{i+1}^{*}(x_k) \le \tilde{V}_{i+1}^{\overline{\varepsilon}}(x_k) \tag{57}$$

Noting that

$$\begin{aligned} 2\tilde{V}_{i+1}^{*}(x_k) &= \underline{\varepsilon}U(x_k, \tilde{v}_i(x_k)) + \overline{\varepsilon}U(x_k, \tilde{v}_i(x_k)) + 2\tilde{V}_i(x_{k+1}) \\ &\ge \tilde{V}_{i+1}^{\underline{\varepsilon}}(x_k) + \tilde{V}_{i+1}^{\overline{\varepsilon}}(x_k) \end{aligned} \tag{58}$$

we can use the following inequality (59) instead of (51) for simplicity

$$|\tilde{V}_{i+1}(x_k) - \tilde{V}_{i+1}^{*}(x_k)| \le \tilde{V}_{i+1}^{\overline{\varepsilon}}(x_k) - \tilde{V}_{i+1}^{*}(x_k) \tag{59}$$

That means if we make (59) hold during iteration, (51) can be satisfied automatically.

Next, we will show the convergence property considering the NN approximation error.

Theorem 4: The sequence $\{\tilde{v}_i\}_0^\infty$ and $\{v_i\}_0^\infty$ satisfy (50) and (10). The sequence $\{\tilde{V}_i\}_0^\infty$ and $\{V_i\}_0^\infty$ satisfy (51) and (11), with $\tilde{V}_0(x_k) = V_0(x_k) = x_k^{\mathrm{T}} P_0 x_k$. Then

$$\underline{\varepsilon}V_i(x_k) \le \tilde{V}_i(x_k) \le \overline{\varepsilon}V_i(x_k), \quad \forall i, x_k \tag{60}$$

Proof: First, we demonstrate the left-hand side of (60) by mathematical induction, that is

$$\underline{\varepsilon}V_i(x_k) \le \tilde{V}_i(x_k), \quad \forall i, x_k \tag{61}$$

When $i = 0$, it is easy to see that $\underline{\varepsilon}V_0(x_k) \le \tilde{V}_0(x_k)$ for $\underline{\varepsilon} \le 1$. Assume that $\underline{\varepsilon}V_i(x_k) \le \tilde{V}_i(x_k)$, $\forall i \ge 0$ and $\forall x_k$. According to (51) and (53), we have

$$\underline{\varepsilon}U(x_k, \tilde{v}_i^{\underline{\varepsilon}}(x_k)) + \tilde{V}_i(x_{k+1}) \le \tilde{V}_{i+1}(x_k)$$

where $x_{k+1} = f(x_k) + g(x_k)\tilde{v}_i^{\underline{\varepsilon}}(x_k)$. So, we have

$$\underline{\varepsilon}U(x_k, \tilde{v}_i^{\underline{\varepsilon}}(x_k)) + \underline{\varepsilon}V_i(x_{k+1}) \le \tilde{V}_{i+1}(x_k)$$

that is

$$\underline{\varepsilon}[U(x_k, \tilde{v}_i^{\underline{\varepsilon}}(x_k)) + V_i(x_{k+1})] \le \tilde{V}_{i+1}(x_k)$$

Considering

$$\min_{u_k}\{U(x_k, u_k) + V_i(x_{k+1})\} \le \{U(x_k, \tilde{v}_i^{\underline{\varepsilon}}(x_k)) + V_i(x_{k+1})\}$$

we have

$$\underline{\varepsilon}V_{i+1}(x_k) \le \tilde{V}_{i+1}(x_k)$$

By mathematical induction, the left-hand side of (60) has been demonstrated.

Similarly, we demonstrate the right-hand side of (60), that is

$$\tilde{V}_i(x_k) \le \overline{\varepsilon}V_i(x_k), \quad \forall i, x_k \tag{62}$$

When $i = 0$, it is easy to see that $\tilde{V}_0(x_k) \le \overline{\varepsilon}V_0(x_k)$ for $\overline{\varepsilon} \ge 1$. Assume that $\tilde{V}_i(x_k) \le \overline{\varepsilon}V_i(x_k)$, $\forall i \ge 0$ and $\forall x_k$. According to (11), we have

$$V_{i+1}(x_k) = U(x_k, v_i(x_k)) + V_i(x_{k+1})$$

where $x_{k+1} = f(x_k) + g(x_k)v_i(x_k)$. So, we have

$$\begin{aligned} \overline{\varepsilon}V_{i+1}(x_k) &= \overline{\varepsilon}[U(x_k, v_i(x_k)) + V_i(x_{k+1})] \\ &\ge \overline{\varepsilon}U(x_k, v_i(x_k)) + \tilde{V}_i(x_{k+1}) \\ &\ge \min_{u_k}\{\overline{\varepsilon}U(x_k, u_k) + \tilde{V}_i(x_{k+1})\} \\ &\ge \tilde{V}_{i+1}(x_k) \end{aligned} \tag{63}$$

Therefore the right-hand side of (60) is also demonstrated. □

Remark 4: According to Theorem 2, we have

$$\begin{aligned} &\underline{\varepsilon}\left[1 + \frac{\alpha - 1}{(1 + \theta^{-1})^i}\right]J^*(x_k) \le \underline{\varepsilon}V_i(x_k) \le \tilde{V}_i(x_k) \\ &\le \overline{\varepsilon}V_i(x_k) \le \overline{\varepsilon}\left[1 + \frac{\beta - 1}{(1 + \theta^{-1})^i}\right]J^*(x_k) \end{aligned} \tag{64}$$

Let $i \to \infty$, we can conclude that

$$\underline{\varepsilon}J^*(x) \le \tilde{V}_\infty(x_k) \le \overline{\varepsilon}J^*(x_k) \tag{65}$$

By the inequality (65), we can make the value function converge to a very small interval containing the optimal cost function. The better approximation ability of NN, that is, the smaller ε, will reduce the interval. In the traditional VI, such conclusion cannot be made, although there always exists NN approximation error.

4 NN implementation of the general VI algorithm

We have demonstrated the convergence of value function in Section 3 under the assumption that the action and value update equations can exactly be solved at each iteration. However, it is difficult to solve these equations for nonlinear systems. Fortunately, we can use NN to approximate v_i and V_i at each iteration. In this section, we will use HDP to implement the general VI algorithm.

The structure diagram of the HDP algorithm is given in Fig. 1. In the HDP algorithm, there are three NNs, which are model network, critic network and action network. The model network is used to approximate the unknown nonlinear system by using available input–output data, which is referred to as a data-based method. The critic network approximates the relationship between state vector x_k and value function $\tilde{V}_i(x_k)$ and the action network approximates the relationship between state vector x_k and control vector $\tilde{v}_i(x_k)$.

We choose the popular BP NN as our function approximation scheme, although any other function approximation structures also suffice. The LM algorithm is used to tune weights of NN instead of gradient descent method. We find that LM algorithm can enormously improve the convergence speed and decrease the approximation error, which will lead ADP to better performance. LM algorithm, which combines steepest descent gradient and Gauss–Newton method, mainly includes three processes: calculating the Jacobian matrix, evaluating whether the parameters are obtaining closer to

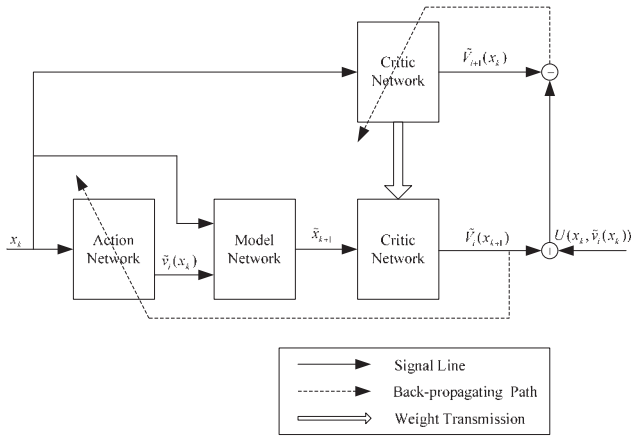

Fig. 1 *Structure diagram of HDP algorithm*

optimal ones or not, and updating the damping parameter. The details of LM algorithm used in the paper can been found in [37, 38, 47].

The first step is to train the model network. The output of model network is denoted as

$$\tilde{x}_{k+1} = \omega_m^{\mathrm{T}}\sigma(\bar{z}_k) = \omega_m^{\mathrm{T}}\sigma(v_m^{\mathrm{T}}z_k) \tag{66}$$

where $z_k = [x_k^{\mathrm{T}}\ \tilde{v}_i^{\mathrm{T}}]^{\mathrm{T}}$ is the input vector of model network. The input to hidden layer weights v_m are a $(n+m)\times l$ matrix and the hidden to output layer weights ω_m are an $l \times n$ matrix, where l is the number of hidden neurons, n is the dimension of state vector and m is the dimension of control input vector. The activation function is chosen as $\sigma(x) = \tanh(x)$ and its derivative is denoted as $\dot{\sigma}(x)$.

The stopping criteria is that the performance function is within prespecified threshold or the training step reaches the maximum value. When the weights of model network converge, they are kept unchanged. Then, the estimated value of the control coefficient matrix $\hat{g}(x_k)$ is given by

$$\hat{g}(x_k) = \frac{\partial(\omega_m^{\mathrm{T}}(k)\sigma(\bar{z}_k))}{\partial u_k} = \omega_m^{\mathrm{T}}(k)\dot{\sigma}(\bar{z}_k)v_m^{\mathrm{T}}(k)\frac{\partial z_k}{\partial u_k} \tag{67}$$

where

$$\frac{\partial z_k}{\partial u_k} = \begin{bmatrix} 0_{n\times m} \\ \cdots \\ I_m \end{bmatrix}$$

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

Similarly, we use LM algorithm to train critic network and action network. The output of the critic network is denoted as

$$\tilde{V}_{i+1}(x_k) = \omega_{c(i+1)}^{\mathrm{T}}\sigma(v_{c(i+1)}^{\mathrm{T}}x_k) \tag{68}$$

The target of value function is given by

$$V_{i+1}(x_k) = x_k^{\mathrm{T}}Qx_k + \tilde{v}_i^{\mathrm{T}}(x_k)R\tilde{v}_i(x_k) + \tilde{V}_i(\tilde{x}_{k+1}) \tag{69}$$

where $\tilde{V}_i(\tilde{x}_{k+1}) = \omega_{c(i)}^{\mathrm{T}}\sigma(v_{c(i)}^{\mathrm{T}}\tilde{x}_{k+1})$. Then, the error function for training critic network is defined by

$$e_{c(i+1)}(x_k) = \tilde{V}_{i+1}(x_k) - V_{i+1}(x_k) \tag{70}$$

and the performance function to be minimised is defined by

$$E_{c(i+1)}(x_k) = \frac{1}{2}e_{c(i+1)}^{\mathrm{T}}e_{c(i+1)} \tag{71}$$

The tuning algorithm of critic network is the same as model network.

In the action network, the state x_k is used as input to obtain the optimal control. The output can be formulated as

$$\tilde{v}_i(x_k) = \omega_{a(i)}^{\mathrm{T}}\sigma\left(v_{a(i)}^{\mathrm{T}}x_k\right) \tag{72}$$

The target of control input is given by

$$v_i(x_k) = -\frac{1}{2}R^{-1}\hat{g}^{\mathrm{T}}(x_k)\frac{\partial\tilde{V}_i(\tilde{x}_{k+1})}{\partial\tilde{x}_{k+1}} \tag{73}$$

where $\tilde{x}_{k+1} = \omega_m^{\mathrm{T}}\sigma(v_m^{\mathrm{T}}[x_k^{\mathrm{T}}\ \tilde{v}_i^{\mathrm{T}}]^{\mathrm{T}})$. The convergence proof of action network weights is given in [33]. The error function of the action network can be defined by

$$e_{a(i)}(x_k) = \tilde{v}_i(x_k) - v_i(x_k) \tag{74}$$

The weights of the action network are updated to minimise the following performance function

$$E_{a(i)}(x_k) = \frac{1}{2}e_{a(i)}^{\mathrm{T}}e_{a(i)} \tag{75}$$

The LM algorithm ensures that $E_{a(i)}(x_k)$ will decrease every time when the parameters of action network update.

At last, a summary of the present optimal control algorithm is given as follows:

Step 1: Initialise the parameters $j_{\max}^m$, $j_{\max}^a$, $j_{\max}^c$, ε_m, ε_a, ε_c $i_{\max}, \xi, Q, R$ and the weights of NN.
Step 2: Construct the model network $\tilde{x}_{k+1} = \omega_m^{\mathrm{T}}\sigma(v_m^{\mathrm{T}}z_k)$. Obtain the training data, and train the model network until the given accuracy ε_m or the maximum number of iterations $j_{\max}^m$ is reached.
Step 3: Set the iteration index $i = 0$ and $P_0 = \gamma I_n$. Choose randomly an array of p state vector $[x_k^1, x_k^2, \ldots, x_k^p]$. Compute the control target $[v_0(x_k^1), v_0(x_k^2), \ldots, v_0(x_k^p)]$ by (73) and train the action network until the given accuracy ε_a or the maximum number of iterations $j_{\max}^a$ is reached.
Step 4: Compute the target of the critic network $[V_1(x_k^1), V_1(x_k^2), \ldots, V_1(x_k^p)]$ by (69). Train the critic network until the given accuracy ε_c or the maximum number of iterations $j_{\max}^c$ is reached.
Step 5: If $V_0 > V_1$ is true for all x_k, go to Step 6; otherwise, increase γ and go to Step 3.
Step 6: Set the iteration index $i = i + 1$. Choose randomly an array of p state vector $[x_k^1, x_k^2, \ldots, x_k^p]$. Compute the target of action network $[v_i(x_k^1), v_i(x_k^2), \ldots, v_i(x_k^p)]$ by (73), and train the action network until the given accuracy ε_a or the maximum number of iterations $j_{\max}^a$ is reached.
Step 7: Compute the output of the action network $[\tilde{v}_i(x_k^1), \tilde{v}_i(x_k^2), \ldots, \tilde{v}_i(x_k^p)]$, the output of the model network $[\tilde{x}_{k+1}^1, \tilde{x}_{k+1}^2, \ldots, \tilde{x}_{k+1}^p]$ and the output of the critic network $[\tilde{V}_i(x_{k+1}^1), \tilde{V}_i(x_{k+1}^2), \ldots, \tilde{V}_i(x_{k+1}^p)]$. Then compute the target of the critic network $[V_{i+1}(x_k^1), V_{i+1}(x_k^2), \ldots, V_{i+1}(x_k^p)]$ by (69). Train the critic network until the given accuracy ε_c or the maximum number of iterations $j_{\max}^c$ is reached.
Step 8: If $i > i_{\max}$ or

$$\|V_{i+1}(x_k^s) - V_i(x_k^s)\|^2 \le \xi, \quad s = 1, 2, \ldots, p$$

go to Step 9; otherwise, go to Step 6.
Step 9: Obtain the final near-optimal control law, and stop.

doi: 10.1049/iet-cta.2011.0783

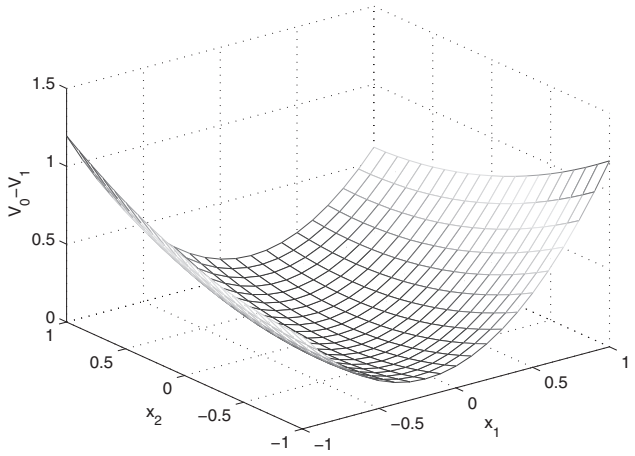

Fig. 2 *3D plot of $V_0 - V_1$ in the operation region*

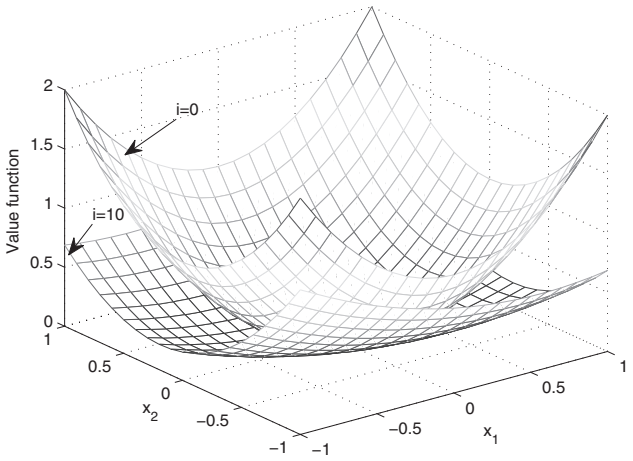

Fig. 3 *3D plot of approximate value function at $i = 0, 10$*

5 Simulation studies

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

Example 1 (discrete-time linear system)*:* Consider the linear system $x_{k+1} = Ax_k + Bu_k$, where

$$A = \begin{bmatrix} 0 & 0.4 \\ 0.3 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{76}$$

The weight matrices are chosen as

$$Q = \begin{bmatrix} 0.2 & 0 \\ 0 & 0.2 \end{bmatrix}, \quad R = 1 \tag{77}$$

Noting that the open-loop poles are $z_1 = -0.1083$ and $z_2 = 1.1083$, the system is unstable.

To reduce the influence of the NN approximation errors, we choose three-layer BP NNs as model network, critic network and action network with the structures 3–9–2, 2–8–1 and 2–8–1, respectively, although a linear NN may also work here. The initial weights of NNs are chosen randomly in $[-0.1, 0.1]$. The inner-loop iteration number of critic network and action network is $j^c_{\max} = j^a_{\max} = 1000$, and the given accuracy is $\varepsilon_c = \varepsilon_a = 10^{-6}$. The maximum outer-loop iteration is selected as $i_{\max} = 10$ and the prespecified accuracy is selected as $\xi = 10^{-6}$. The number of samples at each iteration is $p = 2000$. Before implementing the general VI algorithm, we need to train the model network first. The operation region of the system (1) is selected as $-1 \le x_1 \le 1$ and $-1 \le x_2 \le 1$. Thousand samples are randomly chosen from this operation region as the training set, and the model network is trained until the given accuracy $\varepsilon_m = 10^{-8}$ is reached with $j^m_{\max} = 10000$.

Set $P_0 = I_2$. We find that $V_0 \ge V_1$ holds for all states, which can be seen from Fig. 2. After implementing the outer-loop iteration for ten times, the convergence of value function is observed. The three-dimensional (3D) plot of approximate value function at $i = 0$ and $i = 10$ is given in Fig. 3, and the 3D plot of error between optimal value function J^* and approximate optimal value function V_{10} is given in Fig. 4. We can see that the error between optimal value function and approximate optimal value function is nearly within 10^{-3} in the operational region from Fig. 4.

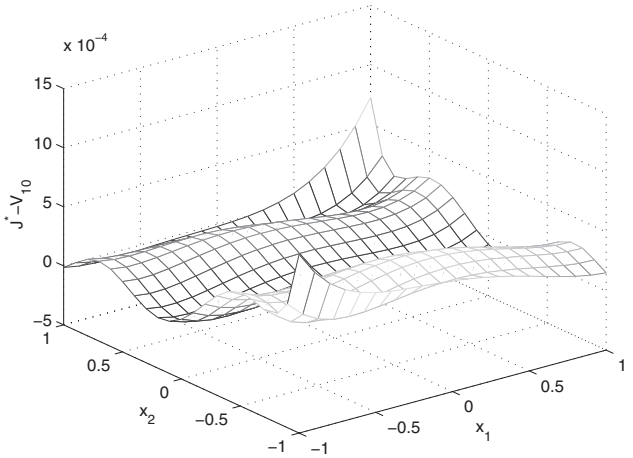

Fig. 4 *Error between optimal value function J^* and approximate value function V_{10}*

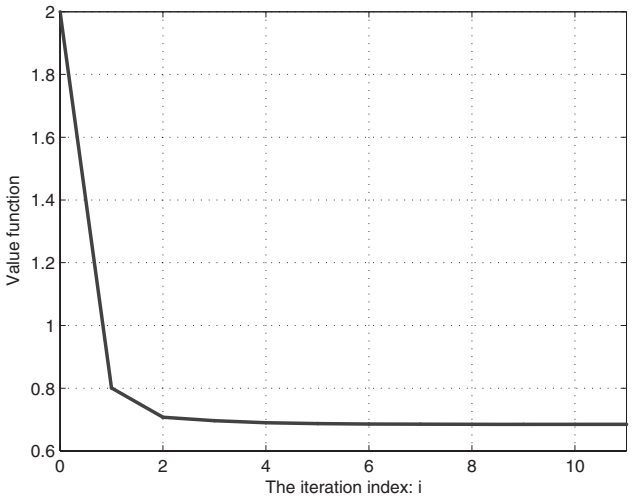

Fig. 5 *Convergence process of the value function at $x = (1, -1)$*

For the initial state $x_0 = [1 \quad -1]^{\mathrm{T}}$, the convergence process of value function is given in Fig. 5. We apply the control law v_{10} to the system for 20 time steps. The corresponding state trajectories are given in Fig. 6, and the control input is shown in Fig. 7.

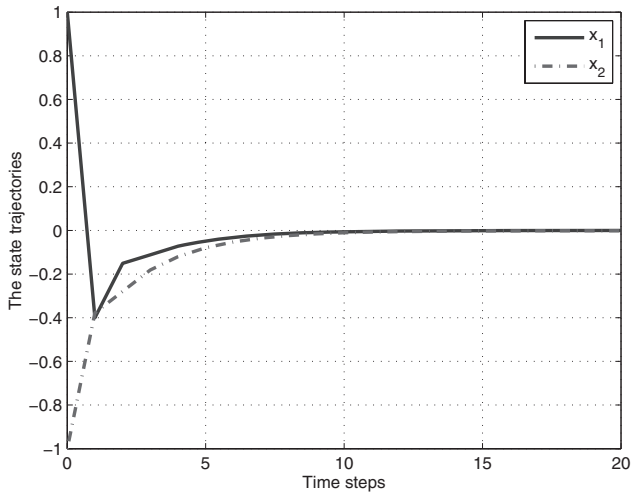

Fig. 6 *State trajectories of Example 1*

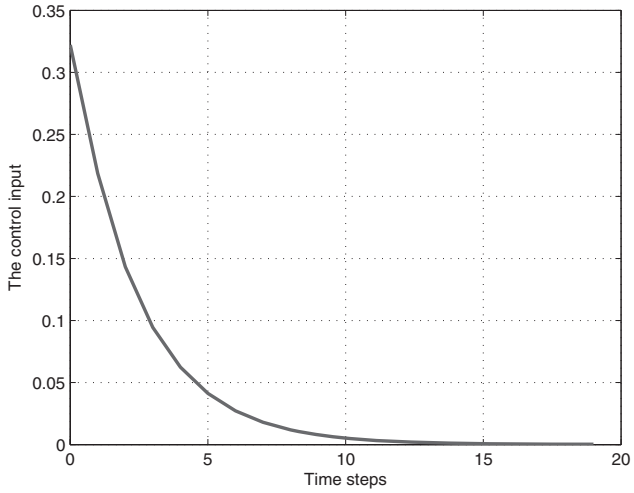

Fig. 7 *Control input of Example 1*

Example 2 (Discrete-time non-linear system): Consider the following discrete-time affine non-linear system $x_{k+1} = f(x_k) + g(x_k)u_k$, where

$$f(x_k) = \begin{bmatrix} 0.9x_{1_k} + 0.1x_{2_k} \\ -0.05(x_{1_k} + x_{2_k}(1 - (\cos(2x_{1_k}) + 2)^2)) + x_{2_k} \end{bmatrix}$$

$$g(x_k) = \begin{bmatrix} 0 \\ 0.1\cos(2x_{1_k}) + 0.2 \end{bmatrix} \tag{78}$$

The cost function is

$$J = \sum_{k=0}^{\infty} \left(x_k^{\mathrm{T}} Q x_k + u_k^{\mathrm{T}} R u_k\right) \tag{79}$$

where $Q = 0.1\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ and $R = 0.1$.

The structures of model network, critic network and action network are chosen as 3–9–2, 2–10–1 and 2–10–1, respectively. The maximum outer-loop iteration is selected as $i_{\max} = 20$ and the prespecified accuracy is selected as $\xi = 10^{-6}$. Other parameters are chosen the same as in Example 1.

We also set $P_0 = I_2$. We find that $V_0 \geq V_1$ holds for all states, which can be seen from Fig. 8. After implementing the outer-loop iteration for 20 times, the convergence

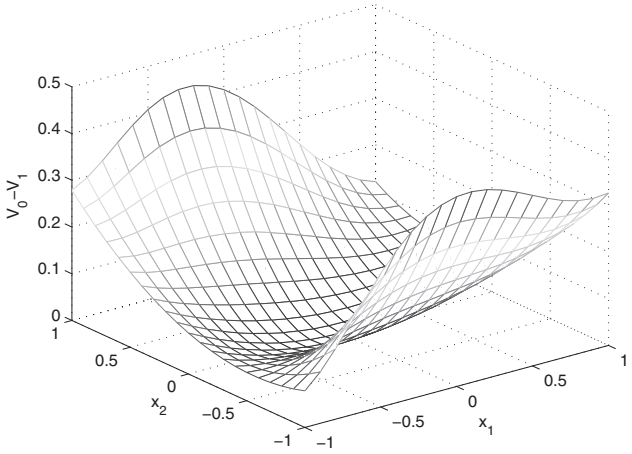

Fig. 8 *3D plot of $V_0 - V_1$ in the operation region*

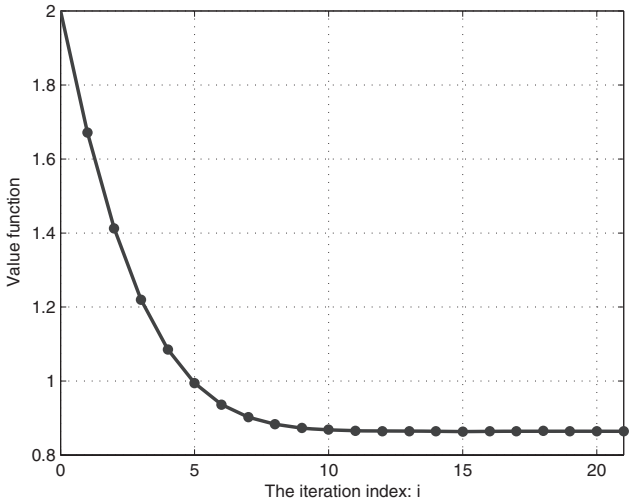

Fig. 9 *Convergence process of the value function at $x = (1, -1)$*

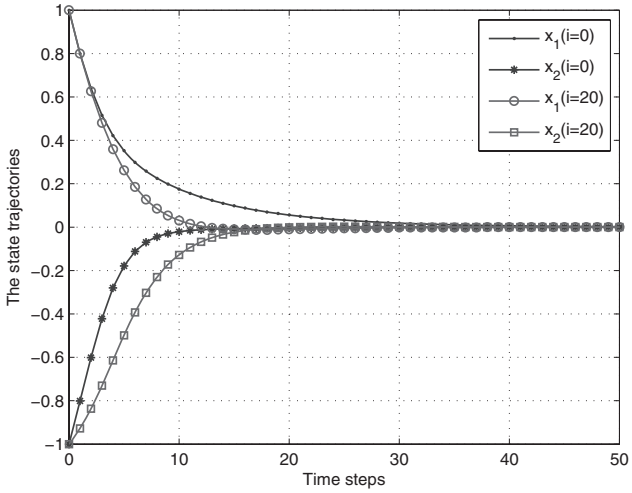

Fig. 10 *State trajectories of Example 2*

of value function is observed. For the initial state $x_0 = [1 \quad -1]^{\mathrm{T}}$, the convergence process of value function is given in Fig. 9. We apply the control law v_{20} to the system for 50 time steps. The corresponding state trajectories are given in Fig. 10, and the control inputs are shown in

doi: 10.1049/iet-cta.2011.0783

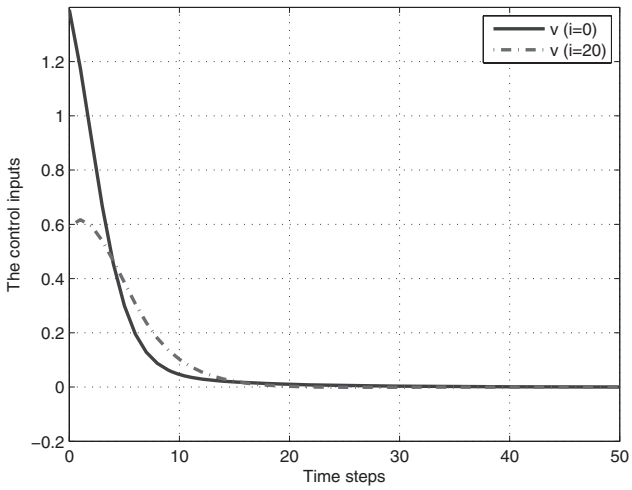

Fig. 11 *Control inputs of Example 2*

Fig. 11. To verify the stability of control law obtained during iteration, we apply the control law v_0 to the system, and the corresponding state trajectories and control inputs are also given in Figs. 10 and 11. We can see that the system with the control law v_0 is stable.

6 Conclusions

In this paper, a novel ADP scheme based on general VI is developed to obtain near optimal control for discrete-time non-linear systems. The value function iteration algorithm is given in a general framework with convergence and stability analysis, and the approximation error of NN is also taken into consideration. Three NNs are used to facilitate the implementation of the iterative algorithm. The effectiveness of the developed scheme is demonstrated by simulation.

7 Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant numbers 60904037, 60921061 and 61034002; in part by the Beijing Natural Science Foundation under Grant no. 4102061; and in part by the China Postdoctoral Science Foundation under Grant no. 201104162.

8 References

1 Bellman, R.E.: 'Dynamic programming' (Princeton University Press, 1957)
2 Bryson, A.E., Ho, Y.C.: 'Applied optimal control: optimization, Estimation, and Control' (Hemisphere-Wiley, 1975)
3 Lewis, F.L., Syrmos, V.L.: 'optimal control' (Wiley, 1995)
4 Souganidi, P.E.: 'Approximation schemes for viscosity solutions of Hamilton–Jacobi equations', *J. Diff. Equ.*, 1985, **59**, pp. 1–43
5 Beard, R., Saridis, G., Wen, J.: 'Galerkin approximations of the generalized Hamilton–Jacobi–Bellman equation', *Automatica*, 1997, **33**, (12), pp. 2158–2177
6 Murray, J.J., Cox, C.J., Lendaris, G.G.: 'Adaptive dynamic programming', *IEEE Trans. Syst. Man Cybern. C, Appl. Rev.*, 2002, **32**, (2), pp. 140–153
7 Abu-Khalaf, M., Lewis, F.L.: 'Nearly optimal control laws for nonlinear systems with saturating actuators using a neural network HJB approach', *Automatica*, 2005, **41**, (5), pp. 779–791
8 Cheng, T., Lewis, F.L., Abu-Khalaf, M.: 'A neural network solution for fixed-final time optimal control of nonlinear systems', *Automatica*, 2007, **43**, (3), pp. 482–490
9 Chen, Z., Jagannathan, S.: 'Generalized Hamilton–Jacobi–Bellman formulation-based neural network control of affine nonlinear discretetime systems', *IEEE Trans. Neural Netw.*, 2008, **19**, (1), pp. 90–106
10 Werbos, P.J.: 'A menu of designs for reinforcement learning over time', in Miller, W.T., Sutton, R.S., Werbos, P.J. (Eds.): 'Neural networks for control', (MIT Press, 1990) Ch. 13
11 Si, J., Barto, A.G., Powell, W.B.: 'Handbook of learning and approximate dynamic programming' (IEEE Press, 2004)
12 Wang, F.Y., Zhang, H., Liu, D.: 'Adaptive dynamic programming: an introduction', *IEEE Computat. Intell. Mag.*, 2009, **4**, (2), pp. 39–47
13 Lewis, F.L., Vrabie, D.: 'Reinforcement learning and adaptive dynamic programming for feedback control', *IEEE Circuits Syst. Mag.*, 2009, **9**, (3), pp. 32–50
14 Prokhorov, D.V., Wunsch, D.C.: 'Adaptive critic designs', *IEEE Trans. Neural Netw.*, 1997, **8**, (5), pp. 997–1007
15 Ding, J., Balakrishnan, S.N.: 'Approximate dynamic programming solutions with a single network adaptive critic for a class of nonlinear systems', *J Control Theory Appl.*, 2011, **9**, (3), pp. 370–380
16 Sutton, R.S., Barto, A.G.: 'Reinforcement learning: an introduction' (MIT Press, 1998)
17 Lewis, F.L., Lendaris, G., Liu, D.: 'Special issue on approximate dynamic programming and reinforcement learning for feedback control', *IEEE Trans. Syst. Man Cybern. B, Cybern.*, 2008, **38**, (4), pp. 896–897
18 He, P., Jagannathan, S.: 'Reinforcement learning neural-network-based controller for nonlinear discrete-time systems with input constraints', *IEEE Trans. Syst. Man Cybern. B, Cybern.*, 2007, **37**, (2), pp. 425–436
19 Yang, Q., Vance, J.B., Jagannathan, S.: 'Control of nonaffine nonlinear discrete-time systems using reinforcement-learning-based linearly parameterized neural networks', *IEEE Trans. Syst. Man Cybern. B, Cybern.*, 2008, **38**, (4), pp. 994–1001
20 Shih, P., Kaul, B.C., Jagannathan, S.: 'Reinforcement-learning-based dual-control methodology for complex nonlinear discrete-time systems with application to spark engine EGR operation', *IEEE Trans. Neural Netw.*, 2008, **19**, (8), pp. 1369–1388
21 Yang, L., Si, J., Tsakalis, K.S., Rodriguez, A.A.: 'Direct heuristic dynamic programming for nonlinear tracking control with filtered tracking error', *IEEE Trans. Syst. Man Cybern. B, Cybern.*, 2009, **39**, (6), pp. 1617–1622
22 Chakraborty, S., Simoes, M.G.: 'Neural dynamic programming based online controller with a novel trim approach', *IEE Proc. Control Theory Appl.*, 2005, **152**, (1), pp. 95–104
23 Lin, W.S., Chang, L.H., Yang, P.C.: 'Adaptive critic anti-slip control of wheeled autonomous robot', *IET Control Theory Appl.*, 2007, **1**, (1), pp. 51–57
24 Li, X.D., Xiao, T.F., Zheng, H.X.: 'Adaptive discrete-time iterative learning control for non-linear multiple input multiple output systems with iteration-varying initial error and reference trajectory', *IET Control Theory Appl.*, 2011, **5**, (9), pp. 1131–1139
25 Meng, D., Jia, Y., Du, J.: 'Robust iterative learning control design for uncertain time-delay systems based on a performance index', *IET Control Theory Appl.*, 2010, **4**, (5), pp. 759–772
26 Howard, R.A.: 'Dynamic Programming and Markov Processes' (MIT Press, 1960)
27 Vrabie, D., Pastravanu, O., Abu-Khalaf, M.: 'Adaptive optimal control for continuous-time linear systems based on policy iteration', *Automatica*, 2009, **45**, (2), pp. 477–484
28 Vrabie, D., Lewis, F.L.: 'Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems', *Neural Netw.*, 2009, **22**, (3), pp. 237–246
29 Vrabie, D., Lewis, F.L.: 'Generalized policy iteration for continuous-time systems', Proc. Int. Joint Conf. Neural Networks, Atlanta, 2009, pp. 3224–3231
30 Vamvoudakis, K.G., Lewis, F.L.: 'Online actor-critic algorithm to solve the continuous-time infinite horizon optimal control problem', *Automatica*, 2010, **46**, (5), pp. 878–888
31 Lewis, F.L., Vamvoudakis, K.G.: 'Reinforcement learning for partially observable dynamic processes: adaptive dynamic programming using measured output data', *IEEE Trans. Syst. Man Cybern. B*, 2011, **41**, (1), pp. 14–24
32 Al-Tamimi, A., Lewis, F.L., Abu-Khalaf, M.: 'Discrete-time nonlinear HJB solution using approximate dynamic programming: convergence proof', *IEEE Trans. Syst. Man Cybern. B, Cybern.*, 2008, **38**, (4), pp. 943–949
33 Dierks, T., Thumati, B.T., Jagannathan, S.: 'Optimal control of unknown affine nonlinear discrete-time systems using offline-trained neural networks with proof of convergence', *Neural Netw.*, 2009, **22**, (5), pp. 851–860

34 Zhang, H., Wei, Q., Luo, Y.: 'A novel infinite-time optimal tracking control scheme for a class of discrete-time nonlinear systems via the greedy HDP iteration algorithm', *IEEE Trans. Syst. Man Cybern. B, Cybern.*, 2008, **38**, (4), pp. 937–942

35 Wang, F., Jin, N., Liu, D., Wei, Q.: 'Adaptive dynamic programming for finite-horizon optimal control of discrete-time nonlinear sytems with ε-error bound', *IEEE Trans. Neural Netw.*, 2011, **22**, (1), pp. 24–36

36 Zhang, H., Luo, Y., Liu, D.: 'Neural-network-based near-optimal control for a class of discrete-time affine nonlinear systems with control constraints', *IEEE Trans. Neural Netw.*, 2009, **20**, (9), pp. 1490–1503

37 Madsen, K., Nielsen, H., Tingleff, O.: 'Methods for non-linear least squares problems', Inf. Math. Model. (DTU, 2004, 2nd edn.)

38 Fu, J., He, H., Zhou, X.: 'Adaptive learning and control for MIMO system based on adaptive dynamic programming', *IEEE Trans. Neural Netw.*, 2011, **22**, (7), pp. 1133–1148

39 Leake, R.J., Liu, R.W.: 'Construction of suboptimal control sequences', *SIAM J. Control*, 1967, **5**, (1), pp. 54–63

40 Lincoln, B., Rantzer, A.: 'Relaxing dynamic programming', *IEEE Trans. Autom. Control*, 2006, **51**, (8), pp. 1249–1260

41 Rantzer, A.: 'Relaxed dynamic programming in switching systems', *IEE Proc. Control Theory Appl.*, 2006, **153**, (5), pp. 567–574

42 Apostol, T.: 'Mathematical analysis' (Addison-Wesley, 1974)

43 Liao, X., Wang, L., Yu, P.: 'Stability of dynamical systems' (Elsevier Press, 2007)

44 Bitmead, R.R., Gever, M., Petersen, I.R.: 'Monotonicity and stabilizability properties of solutions of the Riccati difference equation: propositions, lemmas, theorems, fallacious conjectures and counterexamples', *Syst. Control Lett.*, 1985, **5**, pp. 309–315

45 Zhang, H., Huang, J., Lewis, F.L.: 'An improved method in receding horizon control with updating of terminal cost function', in Valavanis, K.P. (Ed.): 'Applications of intelligent control to engineering systems' (Springer, 2009), pp. 365–393

46 Primbs, J.A., Nevistic, V.: 'Feasibility and stability of constrained finite receding horizon control', *Automatica*, 2000, **36**, (7), pp. 965–971

47 Hagan, M.T., Menhaj, M.B.: 'Training feedforward networks with the Marquardt algorithm', *IEEE Trans. Neural Netw.*, 1994, **5**, (6), pp. 989–993

doi: 10.1049/iet-cta.2011.0783