To appear in the *International Journal of Systems Science*
Vol. 00, No. 00, Month 20XX, 1–17

A Novel Optimal Tracking Control Scheme for a Class of Discrete-Time Nonlinear Systems Using Generalized Policy Iteration Adaptive Dynamic Programming Algorithm

Qiao Lin[a], Qinglai Wei[a]*, Derong Liu[b]

[a]*The State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China*
[b]*School of Automation and Electrical Engineering, University of Science and Technology Beijing, Beijing 100083, China*

(*Received 00 Month 20XX; final version received 00 Month 20XX*)

In this paper, a novel iterative adaptive dynamic programming (ADP) algorithm, called generalized policy iteration ADP algorithm, is developed to solve optimal tracking control problems for discrete-time nonlinear systems. The idea is to use two iteration procedures, including an i-iteration and a j-iteration, to obtain the iterative tracking control laws and the iterative value functions. By system transformation, we first convert the optimal tracking control problem into an optimal regulation problem. Then the generalized policy iteration ADP algorithm, which is a general idea of interacting policy and value iteration algorithms, is introduced to deal with the optimal regulation problem. The convergence and optimality properties of the generalized policy iteration algorithm are analyzed. Three neural networks are used to implement the developed algorithm. Finally, simulation examples are given to illustrate the performance of the present algorithm.

Keywords: adaptive dynamic programming; affine nonlinear systems; discrete-time; generalized policy iteration; neural network; tracking control;

1. Introduction

It is well known that the optimal tracking control problems have always been the key focus of control field in recent decades. The traditional tracking controller designed by feedback linearization technique (Ha and Gilbert 1987) is only effective in the neighborhood of the equilibrium point. To avoid this shortcoming, an effective brain-like method called adaptive/approximate dynamic programming (ADP) (Werbos 1977; Werbos et al. 1991) was proposed. The ADP algorithm, which can overcome the curse of dimensionality problem in dynamic programming, has played an important role in finding the optimal control policy and solving the Hamilton-Jacobi-Bellman (HJB) equation forward-in-time. Therefore a large number of studies on ADP have been published. There are several synonyms used for ADP including adaptive dynamic programming (Liu et al. 2012; Wei et al. 2014; Wang et al. 2012; Wei et al. 2016), adaptive critic designs (Liang et al. 2012; Prokhorov and Wunsch 1997; Xu et al. 2013), approximate dynamic programming (Werbos 1992; Staroswiecki et al. 2004; Al-Tamimi et al. 2008; Luo and Wu 2012), neural dynamic programming (Enns et al. 2003; Zhang et al. 2014), neurodynamic programming (Bertsekas and Tsitsiklis 1996; Luo et al. 2015), and reinforcement learning (Staroswiecki et al. 2004; Bertsekas and Tsitsiklis 1996; Luo et al. 2015). According to Prokhorov and Wunsch (1997) and Werbos (1992), ADP approaches were classified into several schemes including heuristic dynamic programming (HDP), dual heuristic programming (DHP), action dependent HDP (ADHDP), also known as Q-learning (Sutton and Barto 1998), and

*Corresponding author. Email: qinglai.wei@ia.ac.cn

action dependent DHP (ADDHP).

Policy and value iteration algorithms are primary tools in ADP to obtain the solution of the HJB equation indirectly and have received more and more attention (Huang and Liu 2013; Liu and Wei 2013; Wei et al. 2015; Wang et al. 2012; Wei et al. 2016; Zhang et al. 2011). Al-Tamimi et al. (2008) studied a value iteration algorithm for deterministic discrete-time affine nonlinear systems. Zhang et al. (2008) and Wang et al. (2012) used the value iteration algorithm to solve optimal tracking control problems for nonlinear systems. Liu and Wei (2014) proposed data-driven neuro-optimal temperature control of water gas shift reaction using the policy iterative algorithm. Value iteration of HDP was implemented using RBF neural networks (NNs) by Zhang et al. (2009). Liu et al. (2012) realized the value iteration of ADP by globalized DHP (GDHP). Wei et al. (2015) considered optimal multi-battery coordination control for home energy management systems via a distributed value iterative algorithm. Abu-Khalaf and Lewis (2005) proposed a policy iteration algorithm for continuous-time nonlinear systems with control constraints. Zhang et al. (2011) applied the policy iteration algorithm to solve a class of nonlinear zero-sum differential games. Bhasin et al. (2013) proposed an online actor-critic-identifier architecture to approximate the optimal control law for uncertain nonlinear systems by policy iteration algorithms. Zhang et al. (2014) used policy learning algorithm to solve H_∞ state feedback control of unknown affine nonlinear discrete-time systems. However, Sutton and Barto (1998) pointed out that almost all reinforcement learning and ADP methods could be described as generalized policy iteration algorithm. So it's important to investigate the generalized policy iteration algorithm for the development of ADP. Furthermore, as a new method to solve the optimal control problems, the generalized policy iteration algorithm has drawn more and more researchers' attention. Vrabie and Lewis (2009) and Vrabie et al. (2009) considered the generalized policy iteration algorithm for continuous-time systems. The analysis of the generalized policy iteration algorithm for discrete-time systems was discussed by Wei et al. (2015) and Liu et al. (2015). To the best of our knowledge, however, there is still no result to solve the optimal tracking control scheme for discrete-time nonlinear systems using generalized policy iteration ADP algorithm. This motivates our research.

In this paper, a new algorithm, called generalized policy iteration ADP algorithm, is employed to design optimal tracking controller for a class of discrete-time nonlinear systems for the first time. Compared to other iteration algorithms for tracking control systems (Zhang et al. 2008; Wang et al. 2012), the present generalized policy iteration ADP algorithm has two iteration procedures, including an i-iteration and a j-iteration. Moreover, the developed algorithm can avoid solving the HJB equation for i-iteration, which accelerates the convergence rate of the algorithm. First, in order to carry out the newly proposed method, we will transform the tracking control problem into an optimal regulation problem. Second, the detailed iteration procedure of the generalized policy iteration algorithm for discrete-time is presented. Starting with an arbitrary admissible control law, we will prove the convergence criteria of the generalized policy iteration algorithm. When the convergence condition is satisfied, the iterative value function and the control law will both converge to the optimum. NNs are introduced to implement the generalized policy iteration algorithm. Using NNs, we can acquire the approximate optimal control law and performance index function. Then simulation results of two examples will confirm the effectiveness of the generalized policy iteration ADP algorithm.

The rest of this paper is organized as follows. In Section 2, we present the problem statement. In Section 3, the monotonicity and convergence properties of the generalized policy iteration ADP algorithm are provided. In Section 4, the NN implementation for the generalized policy iteration algorithm is studied. In Section 5, two examples are provided to illustrate the effectiveness of the present algorithm. Finally, the conclusions are given in Section 6.

2. Problem statement

In this paper, we consider a class of discrete-time systems which are described by:

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

where $x(k) \in \mathbb{R}^n$, $f(x(k)) \in \mathbb{R}^n$, $g(x(k)) \in \mathbb{R}^{n\times m}$ and the input $u(x(k)) \in \mathbb{R}^m$. Here, we suppose that the system is controllable on $\Omega \subset \mathbb{R}^n$, and the generalized inverse of $g(\cdot)$ exists. For optimal tracking control problem, the control objective is to find an optimal control law $u^*(x(k))$, so as to make the nonlinear systems (1) track the specified desired trajectory $\rho(k) \in \mathbb{R}^n$, where we suppose $\rho(k)$ satisfies $\rho(k+1) = \Gamma(\rho(k))$. For simplicity, $u(x(k))$ is replaced by $u(k)$ in this paper.

The tracking error is defined as

$$e(k) = x(k) - \rho(k). \tag{2}$$

Inspired by the study of Dierks and Jagannathan (2009) and Park et al. (1996), we can define

$$v(k) = u(k) - u_\rho(k), \tag{3}$$

where

$$u_\rho(k) = g^{-1}(\rho(k))(\rho(k+1) - f(\rho(k))). \tag{4}$$

Here, $u_\rho(k)$, which denotes the expected control, is introduced for analytical purpose. By substituting (2), (3) and (4) into (1), we can obtain a new system as follows:

$$\begin{aligned} e(k+1) =& f(e(k)+\rho(k)) + g(e(k)+\rho(k))g^{-1}(\rho(k)) \\ & \times (\rho(k+1) - f(\rho(k))) - \rho(k+1) + g(e(k)+\rho(k))v(k). \end{aligned} \tag{5}$$

As we know, $\rho(k)$ and $\rho(k+1)$ are known in advance, so the new system (5) can be represented as

$$e(k+1) = F(e(k), v(k)), \tag{6}$$

where $e(k)$ is the state vector and $v(k)$ is the control vector. Now let $\underline{v}(k) = \{v(k), v(k+1), v(k+2), \ldots\}$ be an arbitrary sequence of controls from k to ∞. The performance index function for initial state $e(0)$ is defined as

$$J(e(0), \underline{v}(0)) = \sum_{k=0}^{\infty} \{e^{\mathsf{T}}(k)Qe(k) + v^{\mathsf{T}}(k)Rv(k)\}, \tag{7}$$

where $Q \in \mathbb{R}^{n\times n}$, $R \in \mathbb{R}^{m\times m}$ are positive definite matrices and $\underline{v}(0) = \{v(0), v(1), v(2), \ldots\}$. Then, let the utility function U be

$$U(e(k), v(k)) = e^{\mathsf{T}}(k)Qe(k) + v^{\mathsf{T}}(k)Rv(k). \tag{8}$$

In this sense, the nonlinear tracking problem is transformed into a regular optimal control problem. So the goal of this paper is not only to design the optimal control law $v^*(k)$ such that $x(k)$ tracks the desired trajectory $\rho(k)$, but also minimizes the performance index function (7).

According to Bellmans optimality principle, the optimal performance index function

$$J^*(e(k)) = \min_{\underline{v}(k)} J(e(k), \underline{v}(k)) = \min_{\underline{v}(k)} \sum_{i=k}^{\infty} U(e(i), v(i)) \tag{9}$$

can be rewritten as

$$J^*(e(k)) = \min_{v(k)} \{U(e(k), v(k)) + J^*(e(k+1))\}. \tag{10}$$

In other words, $J^*(e(k))$ satisfies the discrete-time HJB (DTHJB) equation. Therefore, the optimal control law can be expressed as

$$v^*(k) = \arg\min_{v(k)} \{U(e(k), v(k)) + J^*(e(k+1))\}. \tag{11}$$

From (8) and (11), we can obtain

$$v^*(k) = -\frac{1}{2} R^{-1} g^{\mathsf{T}}(e(k) + \rho(k)) \frac{\partial J^*(e(k+1))}{\partial e(k+1)}. \tag{12}$$

By using (11) and (12), the DTHJB equation can be rewritten as

$$\begin{aligned} J^*(e(k)) &= U(e(k), v^*(k)) + J^*(e(k+1)) \\ &= e^{\mathsf{T}}(k) Q e(k) + \frac{1}{4} \left(\frac{\partial J^*(e(k+1))}{\partial e(k+1)} \right)^{\mathsf{T}} g(e(k) + \rho(k)) \\ &\quad \times R^{-1} g^{\mathsf{T}}(e(k) + \rho(k)) \frac{\partial J^*(e(k+1))}{\partial e(k+1)} + J^*(e(k+1)). \end{aligned} \tag{13}$$

Generally speaking, the partial derivative of $J^*(e(k+1))$ is difficult to obtain, and therefore leading to the fact that the above DTHJB equation is hard to solve by traditional methods. In order to overcome this difficulty, we will design a novel algorithm to approximate the performance index function and solve the DTHJB equation forward-in-time in the next section.

3. Optimal tracking control based on generalized policy iteration ADP algorithm

3.1. *Derivation of the generalized policy iteration ADP algorithm*

In this subsection, we present the generalized policy iterative ADP algorithm, where the value function and the control law are updated by iterations. Compared with other iterative algorithms for tracking control systems (Wang et al. 2012; Zhang et al. 2008), the developed generalized policy algorithm contains two iteration procedures, which are named i-iteration and j-iteration, respectively.

First, we start with an initial admissible control law $\hat{v}_0(k)$, and let $V_0(e(k))$ satisfy the generalized HJB (GHJB) equation:

$$\begin{aligned} V_0(e(k)) &= U(e(k), \hat{v}_0(k)) + V_0(e(k+1)) \\ &= U(e(k), \hat{v}_0(k)) + V_0(F(e(k), \hat{v}_0(k)). \end{aligned} \tag{14}$$

Then, for $i = 1$, the iterative control law is obtained by

$$\hat{v}_1(k) = \arg\min_{v(k)} \{U(e(k), v(k)) + V_0(F(e(k), v(k)))\}. \tag{15}$$

Let $\{N_1, N_2, N_3, \ldots\}$ be a sequence, where $N_i \geq 0$, $i = 1, 2, 3, \ldots$, are non-negative integers. Let j_1 increase from 0 to N_1, then the value function is updated by

$$V_{1,j_1+1}(e(k)) = U(e(k), \hat{v}_1(k)) + V_{1,j_1}(F(e(k), \hat{v}_1(k))), \tag{16}$$

where

$$V_{1,0}(e(k)) = \min_{v(k)} \{U(e(k), v(k)) + V_0(e(k+1))\}$$
$$= U(e(k), \hat{v}_1(k)) + V_0(F(e(k), \hat{v}_1(k))). \tag{17}$$

We define the iterative value function as

$$V_1(e(k)) = V_{1,N_1}(e(k)). \tag{18}$$

For $i = 2, 3, 4, \ldots$, we can implement the generalized policy iteration ADP algorithm by the following two iteration procedures.

1) i-iteration. By means of this iteration, we can get the control law by

$$\hat{v}_i(k) = \arg\min_{v(k)} \{U(e(k), v(k)) + V_{i-1}(F(e(k), v(k)))\}. \tag{19}$$

2) j-iteration. In this iteration procedure, the iterative value function $V_{i,j_i}(e(k))$ is updated, while the control law remains unchanged:

$$V_{i,j_i+1}(e(k)) = U(e(k), \hat{v}_i(k)) + V_{i,j_i}(F(e(k), \hat{v}_i(k))), \tag{20}$$

where the iteration index j_i increases from 0 to N_i,

$$V_{i,0}(e(k)) = \min_{v(k)} \{U(e(k), v(k)) + V_{i-1}(e(k+1))\}$$
$$= U(e(k), \hat{v}_i(k)) + V_{i-1}(F(e(k), \hat{v}_i(k))) \tag{21}$$

and the iterative value function is given as

$$V_i(e(k)) = V_{i,N_i}(e(k)). \tag{22}$$

In fact, each j-iteration tries to solve the following generalizd HJB (GHJB) equation:

$$V_{i,j_i}(e(k)) = U(e(k), \hat{v}_i(k)) + V_{i,j_i}(F(e(k), \hat{v}_i(k))). \tag{23}$$

Remark 1. In the generalized policy iteration ADP algorithm (14)–(23), i is the iteration index of the control law, j is the iteration index of the value function, and k is the time index. We know that the generalized policy iteration ADP algorithm is a general idea of interacting policy and value iteration algorithms. In fact, when $j = 0$, the developed algorithm becomes a value iteration algorithm (Bertsekas and Tsitsiklis 1996). On the contrary, when $j \to \infty$, the developed algorithm can be considered as a policy iteration (Liu and Wei 2014).

Next, we will prove that the value function and the control law converge to the optimum, i.e., the value function $V_{i,j_i} \to J^*$ and the control law $\hat{v}_i \to v^*$ as $i \to \infty$.

3.2. *Convergence analysis of the generalized policy iteration ADP algorithm*

Theorem 1. *Let the iterative control law $\hat{v}_i(k)$ and the iterative value function $V_{i,j_i}(e(k))$ be obtained by (14)–(23). Then, for $i = 1, 2, \ldots$, $j_i = 0, 1, 2, \ldots, N_i$ and for all $e(k) \in \Omega_e$, the iterative value function $V_{i,j_i}(e(k))$ is a monotonically non-increasing sequence satisfying:*

$$V_{i,j_i+1}(e(k)) \leq V_{i,j_i}(e(k)) \tag{24}$$

and

$$V_{i+1,j_{(i+1)}}(e(k)) \leq V_{i,j_i}(e(k)) \tag{25}$$

where $0 \leq j_i \leq N_i$ *and* $0 \leq j_{i+1} \leq N_{i+1}$.

Proof. The inequality (24) can be proved in two steps by mathematical induction.

Step1: Let $i = 1$. According to (14) and (21), we can obtain

$$\begin{aligned} V_{1,0}(e(k)) &= U(e(k), \hat{v}_1(k)) + V_0(F(e(k), \hat{v}_1(k))) \\ &= \min_{v(k)} \{U(e(k), v(k)) + V_0(F(e(k), v(k)\} \\ &\leq U(e(k), \hat{v}_0(k)) + V_0(F(e(k), \hat{v}_0(k))) \\ &= V_0(e(k)). \end{aligned} \tag{26}$$

Then, using (20) and (26), for $j_1 = 0$, we have

$$\begin{aligned} V_{1,1}(e(k)) &= U(e(k), \hat{v}_1(k)) + V_{1,0}(F(e(k), \hat{v}_1(k))) \\ &\leq U(e(k), \hat{v}_1(k)) + V_0(F(e(k), \hat{v}_1(k))) \\ &= V_{1,0}(e(k)). \end{aligned} \tag{27}$$

By (20) and (27), for $j_1 = 1$, we can obtain that

$$\begin{aligned} V_{1,2}(e(k)) &= U(e(k), \hat{v}_1(k)) + V_{1,1}(F(e(k), \hat{v}_1(k))) \\ &\leq U(e(k), \hat{v}_1(k)) + V_{1,0}(F(e(k), \hat{v}_1(k))) \\ &= V_{1,1}(e(k)). \end{aligned} \tag{28}$$

For $j_1 = s$, where s is a positive integer and $1 < s \leq N_1$, we have

$$\begin{aligned} V_{1,s+1}(e(k)) &= U(e(k), \hat{v}_1(k)) + V_{1,s}(F(e(k), \hat{v}_1(k))) \\ &\leq U(e(k), \hat{v}_1(k)) + V_{1,s-1}(F(e(k), \hat{v}_1(k))) \\ &= V_{1,s}(e(k)). \end{aligned} \tag{29}$$

Therefore (24) holds for $i = 1$.

Step 2: Assuming that (24) holds for $i = m$, we get

$$V_{m,j_m+1}(e(k)) \leq V_{m,j_m}(e(k)). \tag{30}$$

Hence, according to (14) and (21), for $i = m + 1$

$$\begin{aligned} V_{m+1,0}(e(k)) &= U(e(k), \hat{v}_{m+1}(k)) + V_m(F(e(k), \hat{v}_{m+1}(k))) \\ &= \min_{v(k)} \{U(e(k), v(k)) + V_m(F(e(k), v(k)\} \\ &\leq U(e(k), \hat{v}_m(k)) + V_m(F(e(k), \hat{v}_m(k))) \\ &= V_{m,N_m+1}(e(k)) \\ &\leq V_{m,N_m}(e(k)) \\ &= V_m(e(k)). \end{aligned} \tag{31}$$

Next, by observing (20) and (26), for $j_{m+1} = 0$, we have

$$\begin{aligned} V_{m+1,1}(e(k)) &= U(e(k), \hat{v}_{m+1}(k)) + V_{m+1,0}(F(e(k), \hat{v}_{m+1}(k))) \\ &\leq U(e(k), \hat{v}_{m+1}(k)) + V_m(F(e(k), \hat{v}_{m+1}(k))) \\ &= V_{m+1,0}(e(k)). \end{aligned} \tag{32}$$

From (20) and (32), for $j_{m+1} = 1$

$$\begin{aligned} V_{m+1,2}(e(k)) &= U(e(k), \hat{v}_{m+1}(k)) + V_{m+1,1}(F(e(k), \hat{v}_{m+1}(k))) \\ &\leq U(e(k), \hat{v}_{m+1}(k)) + V_{m+1,0}(F(e(k), \hat{v}_{m+1}(k))) \\ &= V_{m+1,1}(e(k)). \end{aligned} \tag{33}$$

Using the same method as (29), for $j_{m+1} = q$, where q is positive integer and $1 < q \leq N_{m+1}$,

$$\begin{aligned} V_{m+1,q+1}(e(k)) &= U(e(k), \hat{v}_{m+1}(k)) + V_{m+1,q}(F(e(k), \hat{v}_{m+1}(k))) \\ &\leq U(e(k), \hat{v}_{m+1}(k)) + V_{m+1,q-1}(F(e(k), \hat{v}_{m+1}(k))) \\ &= V_{m+1,q}(e(k)). \end{aligned} \tag{34}$$

So (24) holds for $i = m + 1$. The mathematical induction is complete.

Next, inequality (25) will be proven. Let $0 \leq j_{i+1} \leq N_{i+1}$. Then according to (22)–(24), we can get

$$V_{i+1}(e(k)) = V_{i+1,N_{i+1}}(e(k)) \leq V_{i+1,j_{i+1}}(e(k)) \leq V_{i+1,0}(e(k)) \leq V_i(e(k)). \tag{35}$$

Therefore the proof of (25) is completed. □

From the inequalities (24) and (25), we can conclude that the iterative value function $V_{i,j_i}(e(k))$ is a monotonically nonincreasing sequence. In the next theorem, we will prove the convergence properties under the premise of the monotonicity property.

Theorem 2. *For $i = 0, 1, 2, ...$, and any $N_i \geq 0$, the iterative value function $V_{i,j_i}(e(k))$, which is obtained by (20), converges to the optimal performance index function $J^*(e(k))$, i.e.,*

$$\lim_{i\to\infty} V_{i,j_i}(e(k)) = J^*(e(k)). \tag{36}$$

Proof. We define $\{V_{i,j_i}(e(k))\} = \{V_0(e(k)), V_{1,0}(e(k)), V_{1,1}(e(k)), \ldots, V_{1,N_1}(e(k)), V_1(e(k)), V_{2,0}(e(k)), V_{2,1}(e(k)), \ldots, V_{2,N_2}(e(k)), \ldots\}$. Then, $\{V_i(e(k))\}$ is selected as a subsequence of $\{V_{i,j_i}(e(k))\}$, i.e., $\{V_i(e(k))\} = \{V_0(e(k)), V_1(e(k)), V_2(e(k)), \ldots\}$. According to Apostol (1974), the sequence $\{V_{i,j_i}(e(k))\}$ and its subsequence $\{V_i(e(k))\}$ will have the same limit, i.e.,

$$\lim_{i\to\infty} V_{i,j_i}(e(k)) = \lim_{i\to\infty} V_i(e(k)). \tag{37}$$

Thus, we can choose to prove the following equation for simplicity,

$$\lim_{i\to\infty} V_i(e(k)) = J^*(e(k)). \tag{38}$$

First, we define the limit of the iterative value function $\{V_i(e(k))\}$, i.e., $V_\infty(e(k)) = \lim_{i\to\infty} V_i(e(k))$. Ac-

cording to Theorem 1 and (21), we have

$$\begin{aligned} V_i(e(k)) &\leq V_{i,0}(e(k)) \\ &= U(e(k), \hat{v}_i(k)) + V_{i-1}(F(e(k), \hat{v}_i(k))) \\ &= \min_{v(k)} \{U(e(k), v(k)) + V_{i-1}(F(e(k), v(k)))\}. \end{aligned} \tag{39}$$

Then, we get

$$\begin{aligned} V_\infty(e(k)) &= \lim_{i\to\infty} V_i(e(k)) \\ &\leq V_i(e(k)) \\ &\leq \min_{v(k)} \{U(e(k), v(k)) + V_{i-1}(F(e(k), v(k)))\}. \end{aligned} \tag{40}$$

Hence, letting $i \to \infty$, we can obtain

$$V_\infty(e(k)) \leq \min_{v(k)} \{U(e(k), v(k)) + V_\infty(F(e(k), v(k)))\}. \tag{41}$$

On the other hand, let $\gamma > 0$ be an arbitrary positive constant. From Theorem 1, we can get that $V_i(e(k))$ is non-increasing sequence, so there exists a positive integer π such that

$$V_\pi(e(k)) - \gamma \leq V_\infty(e(k)) \leq V_\pi(e(k)). \tag{42}$$

Thus, substituting (23) into (42), we can obtain

$$\begin{aligned} V_\infty(e(k)) &\geq U(e(k), \hat{v}_\pi(k)) + V_\pi(F(e(k), \hat{v}_\pi(k))) - \gamma \\ &\geq U(e(k), \hat{v}_\pi(k)) + V_\infty(F(e(k), \hat{v}_\pi(k))) - \gamma \\ &= \min_{v(k)} \{U(e(k), v(k)) + V_\infty(F(e(k), v(k)))\} - \gamma, \end{aligned} \tag{43}$$

which reveals that

$$V_\infty(e(k)) \geq \min_{v(k)} \{U(e(k), v(k)) + V_\infty(F(e(k), v(k)))\}, \tag{44}$$

because of the arbitrariness of γ.

Combining (41) and (44), we can conclude that

$$V_\infty(e(k)) = \min_{v(k)} \{U(e(k), v(k)) + V_\infty(F(e(k), v(k)))\}. \tag{45}$$

Second, according to (10), for any $\xi > 0$, we can find an admissible control sequence $\underline{\omega}(k)$ that satisfies

$$J(e(k), \underline{\omega}(k)) \leq J^*(e(k)) + \xi. \tag{46}$$

Now, we suppose that the length of the control sequence $\underline{\omega}(k)$ is θ. Then using (7), (21) and Theorem 1, we

can obtain

$$\begin{aligned} V_\infty(e(k)) &\leq V_\theta(e(k)) \\ &\leq \min_{v(k)} \{U(e(k), v(k)) + V_{\theta-1}(F(e(k), v(k)))\} \\ &\leq J(e(k), \underline{\omega}(k)). \end{aligned} \tag{47}$$

Combining (46) with (47), we can get

$$V_\infty(e(k)) \leq J^*(e(k)) + \xi, \tag{48}$$

where ξ is arbitrary. Then, we have

$$V_\infty(e(k)) \leq J^*(e(k)). \tag{49}$$

On the other hand, from the definition of $J^*(e(k))$ in (9), for $i = 0, 1, 2, ..., V_i(e(k)) \geq J^*(e(k))$ holds for all $e(k) \in \Omega_e$. Letting $i \to \infty$, we can acquire $V_\infty(e(k)) \geq J^*(e(k))$. Therefore, we have (37) holds. □

From Theorems 1 and 2, we can conclude that the value function sequence $\{V_{i,j_i}(e(k))\}$ is a non-increasing sequence, and converges to the optimal performance index function $J^*(e(k))$, i.e., $V_{i,j_i} \to J^*$ as $i \to \infty$. On the basis of the definition of $v^*(k)$ in (11), it's not difficult to find that when $V_{i,j_i} \to J^*$, $\hat{v}_i \to v^*$ also holds as $i \to \infty$.

3.3. *The details of the generalized policy iteration ADP algorithm for optimal tracking control*

The process of the generalized policy iteration ADP algorithm is described in Algorithm 1.

4. NN implementation of the generalized policy iteration ADP algorithm

It's known that NN is an effective tool to approximate nonlinear functions. In this paper, three NNs, which are called model network, critic network, and action network respectively, are used to implement the algorithm and approximate $V_{i,j_i}(e(k))$ and $\hat{v}_i(k)$. All the NNs are chosen as three-layer back-propagation (BP) networks. The structure diagram of the generalized policy iterative ADP algorithm is shown in Figure 1.

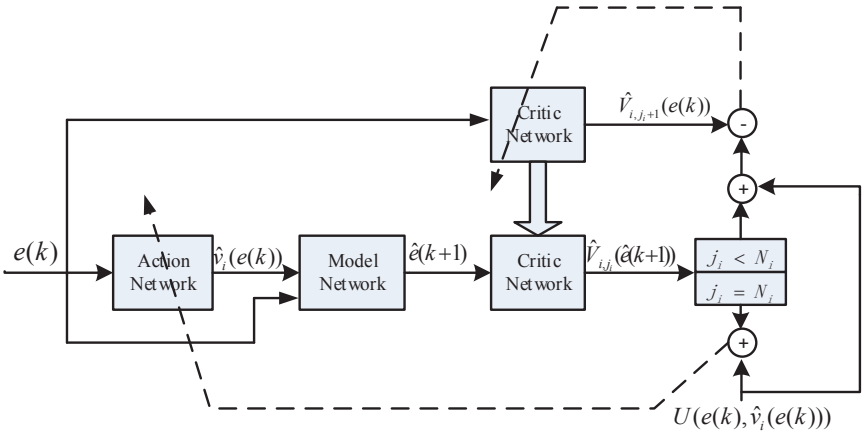


Figure 1. Structure diagram of the algorithm

Algorithm 1 Generalized policy iteration ADP algorithm for optimal tracking control

Step 1	Choose a computation precision ε, the reference trajectory $\rho(k)$, the matrices Q and R, and a positive integer sequence $\{N_i\}$.
Step 2	According to (2) and (5), compute $e(k)$ and transform the tracking problem into a regulation problem.
Step 3	According to the method of Liu and Wei (2014), obtain an initial admissible control law $\hat{v}_0(k)$.
Step 4	Set $i = 1$. Obtain $V_{1,0}(e(k))$ and $\hat{v}_1(k)$ according to algorithms given by Liu et al. (2015).
Step 5	Let j_1 increase from 0 to N_1, and update the iterative value function $V_{1,j_1}(e(k))$ by (16).
Step 6	Set $i = i + 1$. Obtain the control $\hat{v}_i(k)$ by (19).
Step 7	Let j_i increase from 0 to N_i, and update the iterative value function $V_{i,j_i}(e(k))$ by (20).
Step 8	If $\left\|V_{i,N_i}(e(k)) - V_{i-1,N_{i-1}}(e(k))\right\| \leq \varepsilon$, then go to Step 9; otherwise, go to Step 6.
Step 9	Obtain the optimal control law $v^*(k) = \hat{v}_i(k)$.
Step 10	According to (3) and (4), compute the optimal tracking control input $u(k)$ for the system (1).
Step 11	Stop and obtain the optimal tracking controller.

4.1. *The model network*

Before implementing the generalized policy iteration ADP algorithm, the model network should be trained. From the Figure 1, we can obtain the input of model network, including $e(k)$ and $\hat{v}_i(k)$. The output of the model network can be expressed as

$$\hat{e}(k+1) = \omega_m^{\mathsf{T}} \sigma(\eta_m^{\mathsf{T}} p(k)), \tag{50}$$

where $p(k) = [e^{\mathsf{T}}(k), \hat{v}_i^{\mathsf{T}}(k)]^{\mathsf{T}}$, and $\sigma(\cdot)$ is a sigmoid function. The error function of the model network is given as $e_{mk} = \hat{e}(k+1) - e(k+1)$. The objective function to be minimized for the model network is $E_{mk} = \frac{1}{2} e_{mk}^{\mathsf{T}} e_{mk}$.

Then, the weights of the model network can be updated by the gradient-based adaptation as

$$\omega_m(j+1) = \omega_m(j) - \alpha_m \left[\frac{\partial E_{mk}}{\partial \omega_m(j)}\right], \tag{51}$$

$$\eta_m(j+1) = \eta_m(j) - \alpha_m \left[\frac{\partial E_{mk}}{\partial \eta_m(j)}\right], \tag{52}$$

where α_m is the learning rate of the model network, and j denotes the index for updating the weight parameters. Here, we should point out that the weights of the model network should remain unchanged after the training process is finished.

4.2. *The critic network*

The role of the critic network is to approximate the performance index function $V_{i,j_i}(e(k))$. For all $e(k) \in \Omega_e$, the output of the critic network is given as

$$\hat{V}_{i,j_i}(e(k)) = \omega_{c(i,j_i)}^{\mathsf{T}} \sigma(\eta_{c(i,j_i)}^{\mathsf{T}} e(k)). \tag{53}$$

The training target of the value function is obtained by (20). Then, we define the error function of the critic network as $e_{c(i,j_i)k} = \hat{V}_{i,j_i}(e(k)) - V_{i,j_i}(e(k))$. The weights of the critic network are updated to minimize the following performance error measure $E_{c(i,j_i)k} = \frac{1}{2} e_{c(i,j_i)k}^{\mathsf{T}} e_{c(i,j_i)k}$.

So weight updating algorithm of the critic network is expressed as

$$\begin{aligned}
\omega_{c(i,j_i)}(\tau+1) &= \omega_{c(i,j_i)}(\tau) - \alpha_c \left[\frac{\partial E_{c(i,j_i)k}}{\partial \omega_{c(i,j_i)}(\tau)} \right] \\
&= \omega_{c(i,j_i)}(\tau) - \alpha_c \left[\frac{\partial E_{c(i,j_i)k}}{\partial \hat{V}_{i,j_i}(e(k))} \frac{\partial \hat{V}_{i,j_i}(e(k))}{\partial \omega_{c(i,j_i)}(\tau)} \right] \\
&= \omega_{c(i,j_i)}(\tau) - \alpha_c e_{c(i,j_i)k} \left[\frac{\partial \hat{V}_{i,j_i}(e(k))}{\partial \omega_{c(i,j_i)}(\tau)} \right],
\end{aligned} \tag{54}$$

where α_c is the learning rate of the critic network, and τ is the index for updating the weight parameters. The input-hidden weight matrix $\eta_{c(i,j_i)}$ can be updated as same as (54). What we need to do is to replace $\omega_{c(i,j_i)}$ with $\eta_{c(i,j_i)}$.

4.3. *The action network*

In the action network, $e(k)$ is used as the input to design the iterative control law $\hat{v}_i(k)$. The output can be formulated as

$$\underline{\hat{v}}_i(k) = \omega_{ai}^{\mathsf{T}} \sigma(\eta_{ai}^{\mathsf{T}}(e(k))). \tag{55}$$

Then the target of the action network training can be written as

$$\hat{v}_i(k) = \arg\min_{v(k)} \{U(e(k), v(k)) + \hat{V}_{i-1}(F(e(k), v(k)))\}. \tag{56}$$

Define the performance error measure as

$$E_{aik} = \frac{1}{2} e_{aik}^{\mathsf{T}} e_{aik}, \tag{57}$$

where e_{aik} is the error function of the action network, that is

$$e_{aik} = \underline{\hat{v}}_i(k) - \hat{v}_i(k). \tag{58}$$

Similarly, the gradient-based weight updating rule of the action network can be described by

$$\omega_{ai}(\lambda+1) = \omega_{ai}(\lambda) - \alpha_a \left[\frac{\partial E_{aik}}{\partial \omega_{ai}(\lambda)}\right] \tag{59}$$

$$= \omega_{ai}(\lambda) - \alpha_a \left[\frac{\partial E_{aik}}{\partial e_{aik}} \frac{\partial e_{aik}}{\partial \underline{\hat{v}}_i(k)} \frac{\partial \underline{\hat{v}}_i(k)}{\partial \omega_{ai}(\lambda)}\right], \tag{60}$$

where α_a is the learning rate of the action network, and λ is the index for updating the weight parameters. The input-hidden weight matrix η_{ai} can be updated by replacing the ω_{ai} with η_{ai} in (59).

5. Simulation studies

In this section, two simulation examples are given to demonstrate the effectiveness of the generalized policy iteration algorithm for optimal tracking control problems.

Example 1. The first example is given by Heydari and Balakrishnan (2014) with some modifications. Consider the following discrete-time nonlinear system:

$$x(k+1) = f(x(k)) + g(x(k))u(k), \tag{61}$$

where

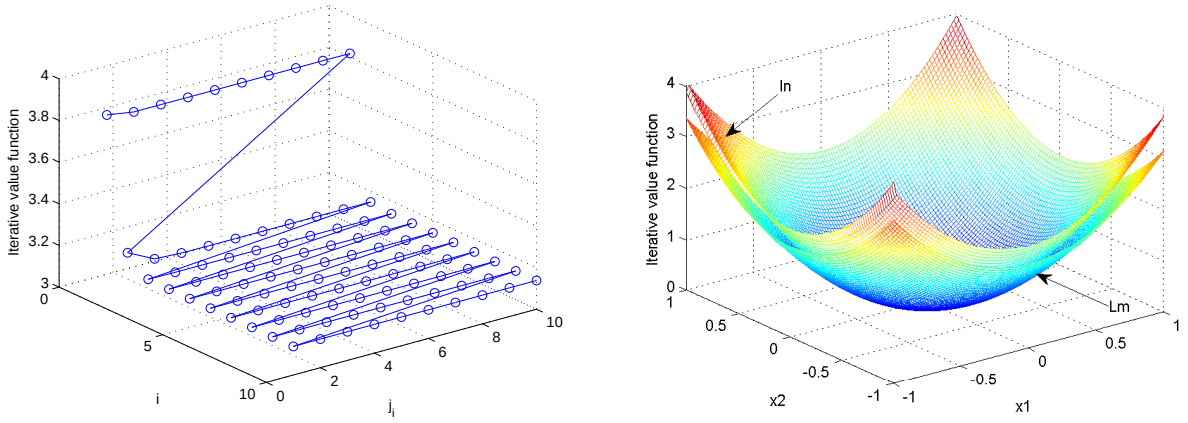
$$x(k) = [x_1(k), x_2(k)]^{\mathsf{T}},$$
$$u(k) = [u_1(k), u_2(k)]^{\mathsf{T}},$$
$$f(x(k)) = \begin{bmatrix} x_1(k) + 0.1x_2(k) \\ -0.1x_1(k) + 1.1x_2(k) - 0.1x_2(k)x_1^2(k) \end{bmatrix},$$
$$g(x(k)) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

Let the initial state be $x(0) = [0.7, -1]^{\mathsf{T}}$ and the desired trajectory is specified as

$$\rho(k) = \left[-\frac{1}{\pi}\cos(0.1\pi k), \sin(0.1\pi k)\right]^{\mathsf{T}}.$$

Then, let the performance index function be expressed by (9). The parameters of the utility function are chosen as $Q = I_1, R = I_2$, where I_1 and I_2 denote the identity matrix with suitable dimensions. The error bound of the iterative ADP is chosen as $\varepsilon = 10^{-5}$. NNs are used to carry out the generalized policy iteration ADP algorithm and the structure of the algorithm is shown in Figure 1. The model network, critic network and action network are chosen as three-layer BP NNs with the structures $4-8-2$, $2-8-1$, $2-8-2$, respectively. All the initial weights are chosen in $[-1, 1]$ randomly. It should be noted that the model network should be trained first. The model network is trained under the learning rate $\alpha_m = 0.15$. When the model network training process is finished, all the weights of the model network are kept unchanged. Then, for each iteration step, the critic network and action network are trained for 1500 times using the learning rate of $\alpha_c = \alpha_a = 0.05$, so that the NN training error become less than ε.

We let iteration index $i_{max} = 10$ and choose the iteration sequence $\{N_i\} = 10$. The changing curve of V_{i,j_i} for $k = 0$ is shown in Figure 2(a) and the trajectory of the iterative value function V_i for the entire state space is shown as Figure 2(b). From Figure 2, we can get that both the value function $V_{i,j_i}(e(k))$ and the subsequence $V_i(e(k))$ are monotonically nonincreasing sequences, where "In" indicates initial iteration and "Lm" means limiting iteration.



(a) V_{i,j_i} for $k = 0$ (b) V_i for the entire state space

Figure 2. Iterative value function

Then, we compute the tracking control law using (19) and apply it to the system (61) for 30 time steps. The state trajectories and the reference trajectories are shown in Figure 3(a) and Figure 3(b), respectively from which we can evaluate the tracking performance. The tracking control curves are shown in Figure 3(c). Figure 3(d) shows the tracking error trajectory. From Figure 3(d), we can see that the tracking errors become minimum, which shows that the control system has already tracked the reference trajectories within the allowable error. These simulation results confirm the excellent performance of the generalized policy iteration algorithm for optimal tracking control systems.

Example 2. The second example is given by Wang et al. (2012). Consider the affine nonlinear discrete-time system given as (61) where

$$f(x(k)) = \begin{bmatrix} 0.2x_1(k)\exp(x_2^2(k)) \\ 0.3x_2^3(k) \end{bmatrix},$$

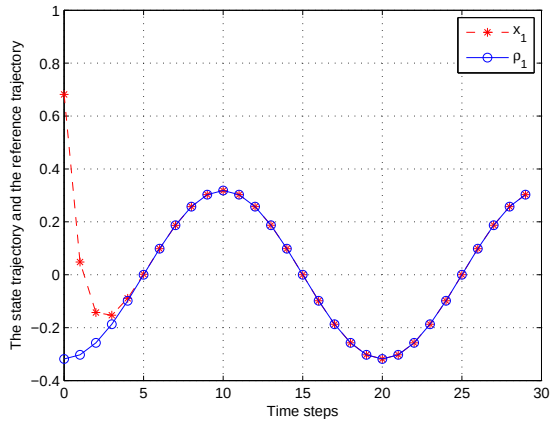
$$g(x(k)) = \begin{bmatrix} -0.25 & 0 \\ 0 & -0.25 \end{bmatrix}.$$

Let the initial state be $x(0) = [1, -0.5]^{\mathsf{T}}$ and the desired trajectory is set to

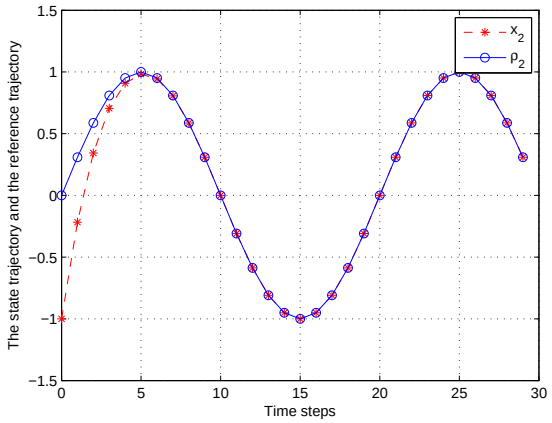
$$\rho(k) = \left[\cos\left(0.5k + \frac{\pi}{2}\right), 0.5\cos(0.5k)\right]^{\mathsf{T}}.$$

The performance index function and the utility function is chosen as same as Example 1 with $Q = I_1$ and $R = I_2$. NNs are also used to carry out the present generalized policy iteration ADP algorithm, where the initial weights and structures of three networks are the same as the ones in Example 1. We take 1000 groups of sampling data, which are chosen from $[-1, 1]$, to train the network. For each NN, the learning rate is chosen as $\alpha_m = \alpha_c = \alpha_a = 0.01$. The networks are trained for 15 iterations, and each iteration includes 4000 training steps to make sure the training error can reach the given bound $\varepsilon = 10^{-5}$.

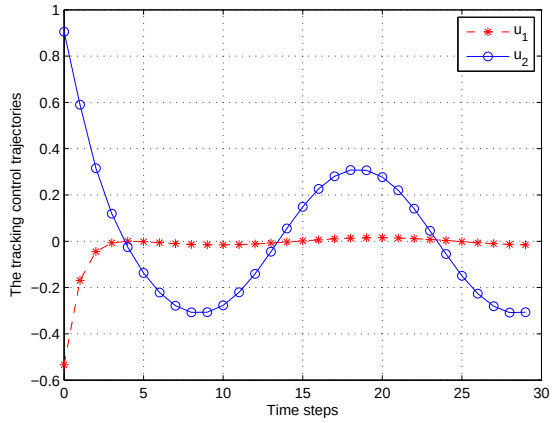
The convergence process of the value function of the generalized policy iteration ADP algorithm is shown in Figure 4, which represents the change of the value function of a specific point $k = 0$. Then, Figures 5(a) and 5(b) show the state trajectories and the reference trajectories. From Figures 5(c) and 5(d), we can acquire the tracking control curves and the tracking error trajectory. Hence, by the simulation results, we can see that the generalized policy iteration ADP algorithm is effective in solving the optimal tracking control problems.



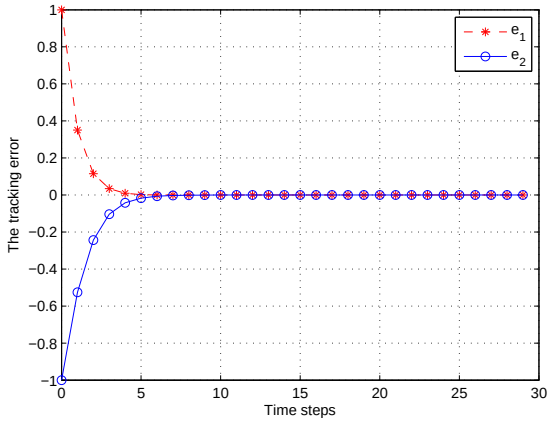
(a) The state trajectory x_1 and the reference trajectory ρ_1



(b) The state trajectory x_2 and the reference trajectory ρ_2



(c) The tracking control input u



(d) The tracking error e

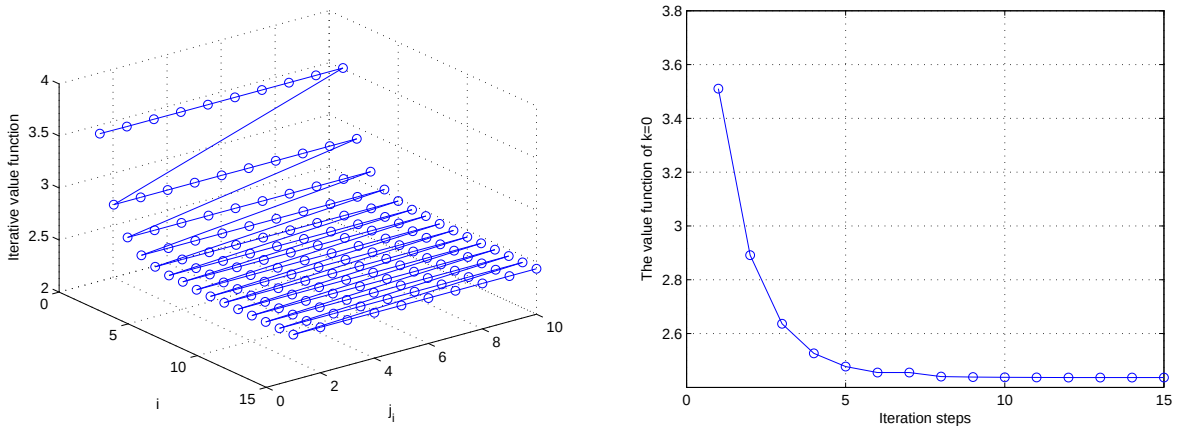
Figure 3. The simulation trajectories in Example 1

6. Conclusion

In this paper, an effective generalized policy iteration ADP algorithm is proposed to solve the optimal tracking control problem. It has been proven that the iterative value functions are monotonically non-increasing and convergent to the optimum. NNs, which can approximate the nonlinear system, control law, and value function, are employed to implement the present algorithm. Finally, two examples are given to verify the effectiveness of the novel algorithm for tracking control systems. However, until now, if the generalized policy iteration algorithm is used to design the optimal tracking controller, the complete knowledge of the system dynamics should be known beforehand. Nevertheless, it is difficult to acquire the complete knowledge in real world. Hence, our future work is to study model-free schemes.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under Grants 61233001, 61273140, 61304086, 61374105, and 61533017 in part by Beijing Natural Science Foundation under Grant 4132078, and in part by the Early Career Development Award of SKLMCCS.



(a) V_{i,j_i} for $k = 0$ (b) V_i for $k = 0$

Figure 4. Iterative value function

References

Abu-Khalaf, M., and Lewis, F. L. (2005). Nearly optimal control laws for nonlinear systems with saturating actuators using a neural network HJB approach. *Automatica*, 41(5), 779–791.

Al-Tamimi, A., Lewis, F. L., Abu-Khalaf, M. (2008). Discrete-time nonlinear HJB solution using approximate dynamic programming: convergence proof. *IEEE Transactions on Systems, Man, and Cybernetics, Part B, Cybernetics*, 38, 943–949.

Apostol, T. M. (1974). *Mathematical Analysis* (2nd ed). Boston, MA, USA: Addison-Wesley Press.

Bertsekas, D. P., and Tsitsiklis, J. N. (1996). *Neuro-Dynamic Programming*, Belmont, MA, USA: Athena Scientific.

Bhasin, S., Kamalapurkar, R., Johnson, M., Vamvoudakis, K. G., Lewis, F. L., and Dixon, W. E. (2013). A novel actor-critic-identifier architecture for approximate optimal control of uncertain nonlinear systems. *Automatica*, 49(1), 82–92.

Dierks, T., and Jagannathan, S. (2009). Optimal trackingc ontrol of affine nonlinear discrete-time systems with unknown internal dynamics. in *Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference*, Shanghai, China, 6750–6755.

Enns, R., and Si, J. (2003). Helicopter trimming and tracking control using direct neural dynamic programming. *IEEE Transactions on Neural networks*, 14(4), 929–939.

Ha, I. J., and Gilbert, E. G. (1987). Robust tracking in nonlinear systems. *IEEE Transactions on Automatic Control*, 32(9), 763–771.

Heydari, A., and Balakrishnan, S. N. (2014). Fixed-final-time optimal tracking control of input-affine nonlinear systems. *Neurocomputing*, 129, 528–539.

Huang, T., and Liu, D. (2013). A self-learning scheme for residential energy system control and management. *Neural Computing & Applications*, 22(2), 259–269.

Liang, J., Venayagamoorthy, G. K., and Harley, R. G. (2012). Wide-area measurement based dynamic stochastic optimal power flow control for smart grids with high variability and uncertainty. *IEEE Transactions on Smart Grid*, 3(1), 59–69.

Liu, D., Wang, D., Zhao, D., Wei, Q., and Jin, N. (2012). Neural-network-based optimal control for a class of unknown discrete-time nonlinear systems using globalized dual heuristic programming. *IEEE Transactions on Automation Science and Engineering*, 9(3), 628–634.

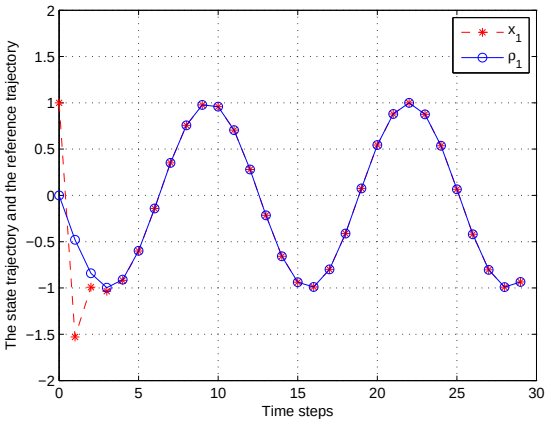
Liu, D., and Wei, Q. (2013). Finite-approximation-error-based optimal control approach for discrete-time nonlinear systems. *IEEE Transactions on Cybernetics*, 43(2), 779–789.

Liu, D., Wei, Q., Yan, P. (2015). Generalized Policy Iteration Adaptive Dynamic Programming for Discrete-Time Nonlinear Systems. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2015.

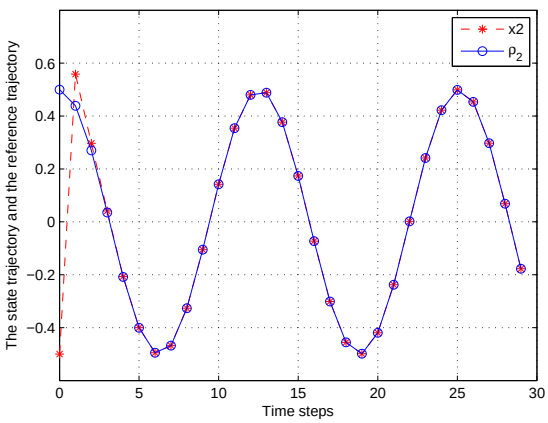
Liu, D., and Wei, Q. (2014). Policy iteration adaptive dynamic programming algorithm for discrete-time nonlinear systems. *IEEE Transactions on Neural Networks and Learning Systems*, 25(3), 621–634.

Liu, D., and Wei, Q. (2014). Data-driven neuro-optimal temperature control of water gas shift reaction using stable iterative adaptive dynamic programming. *IEEE Transactions on Industrial Electronics*, 61(11), 6399–6408.

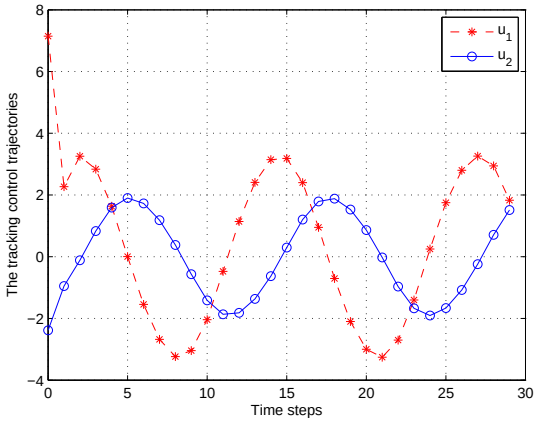
Luo, B., Wu, H., Huang, T. (2015). Off-policy reinforcement learning for H_∞ control design. *IEEE Transactions on*



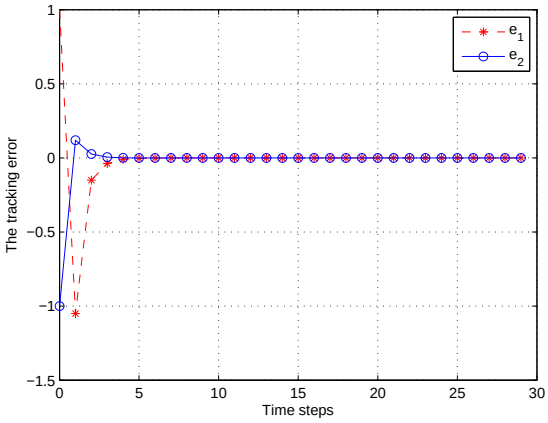
(a) The state trajectory x_1 and the reference trajectory ρ_1



(b) The state trajectory x_2 and the reference trajectory ρ_2



(c) The tracking control input u



(d) The tracking error e

Figure 5. The simulation trajectories in Example 2

Cybernetics, 45(1), 65–76.

Luo, B., Wu, H., Li, H. (2015). Adaptive optimal control of highly dissipative nonlinear spatially distributed processes with neuro-dynamic programming. *IEEE Transactions on Neural Networks and Learning Systems*, 26(4), 684–696.

Luo, B., and Wu, H. (2012). Approximate optimal control design for nonlinear one-dimensional parabolic PDE systems using empirical eigenfunctions and neural network. *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics*, 42(6), 1538–1549.

Park, Y. M., Choi, M. S., Lee, K. Y. (1996). An optimal tracking neuro-controller for nonlinear dynamic systems. *IEEE Transactions on Neural Networks*, 7, 1099–1110.

Prokhorov, D. V., and Wunsch, D. C. (1997). Adaptive critic designs. *IEEE transactions on Neural Networks*, 8(5), 997–1007.

Si, J., Barto, A. G., Powell, W. B., Wunsch, D. C. (2004). Handbook of Learning and Approximate Dynamic Programming. IEEE Press, Wiley, New York.

Sutton, R. S., and Barto, A. G. (1998). *Reinforcement Learning: An Introduction*, MA: MIT Press.

Vrabie, D., and Lewis, F. L. (2009). Generalized policy iteration for continuoustime systems. *Proceedings of International Joint Conference on Neural Networks*, Atlanta, Georgia, USA , 3224–3231.

Vrabie, D., Vamvoudakis, K., and Lewis, F. L. (2009). Adaptive optimal controllers based on generalized policy iteration in a continuous-time framework. in *17th Mediterranean Conference on Control & Automation*, Thessaloniki, Greece, 1402–1409.

Wang, D., Liu, D., Wei, Q., Zhao, D., and Jin, N. (2012). Optimal control of unknown nonaffine nonlinear discrete-time systems based on adaptive dynamic programming. *Automatica*, 48(8), 1825–1832.

Wang, D., Liu, D., and Wei, Q. (2012). Finite-horizon neuro-optimal tracking control for a class of discrete-time nonlinear systems using adaptive dynamic programming approach. *Neurocomputing*, 78(1), 14–22.

Werbos,P. J. (1977). Advanced forecasting methods for global crisis warning and models of intelligence. *General*

Systems Yearbook, 22, 25–38.

Werbos, P. J., Miller, W. T., Sutton, R. S. (1991). A menu of designs for reinforcement learning over time. *Neural Networks for Control*, Cambridge, MA, USA: MIT Press.

Werbos, P. J. (1992). Approximate dynamic programming for real-time control and neural modeling. *Handbook of Intelligent Control* (Chapter 13), Van Nostrand Reinhold, New York.

Wei, Q., Wang, F., Liu, D., and Yang, Q. (2014). Finite-approximation-error based discrete-time iterative adaptive dynamic programming. *IEEE Transactions on Cybernetics*, 44(12), 2820–2833.

Wei, Q., Liu, D., and Lin, Q. (2016) Discrete-time local iterative adaptive dynamic programming: Terminations and admissibility analysis. *IEEE Transactions on Neural Networks and Learning Systems*, accept.

Wei, Q., Liu, D., and Yang, X. (2015). Infinite horizon self-learning optimal control of nonaffine discrete-time nonlinear systems. *IEEE Transactions on Neural Networks and Learning Systems*, 26(4), 866–879.

Wei, Q., Song, R., and Yan, P. (2016). Data-driven zero-sum neuro-optimal control for a class of continuous-time unknown nonlinear systems with disturbance using ADP. *IEEE Transactions on Neural Networks and Learning Systems*, 27(2), 444–458.

Wei, Q., Liu, D., Yang, X. (2015). Infinite Horizon Self-Learning Optimal Control of Nonaffine Discrete-Time Nonlinear Systems. *IEEE Transactions on Neural Networks and Learning Systems*, 26(4), 866–879.

Wei, Q., Liu, D., Shi, G., and Liu, Y. (2015). Optimal multi-battery coordination control for home energy management systems via distributed iterative adaptive dynamic programming. *IEEE Transactions on Industrial Electronics*, 42(7), 4203–4214.

Xu, X., Hou, Z., Lian, C., and He, H. (2013). Online learning control using adaptive critic designs with sparse kernel machines. *IEEE Transactions on Neural Networks and Learning Systems*, 24(5), 762–775.

Zhang, H., Song, R., Wei, Q., and Zhang, T. (2011). Optimal tracking control for a class of nonlinear discrete-time systems with time delays based on heuristic dynamic programming. *IEEE Transactions on Neural Networks*, 22(12), 1851–1862.

Zhang, H., Wei, Q., and Luo, Y. (2008). A novel infinite-time optimal tracking control scheme for a class of discrete-time nonlinear systems via the greedy HDP iteration algorithm. *IEEE Transactions on Systems, Man, and Cybernetics*, 38(4), 937–942.

Zhang, H., Luo, Y., and Liu, D. (2009). The RBF neural network-based near-optimal control for a class of discrete-time affine nonlinear systems with control constraint. *IEEE Transactions on Neural Networks*, 20(9), 1490–1503.

Zhang, H., Wei, Q., and Liu, D. (2011). An iterative adaptive dynamic programming method for solving a class of nonlinear zero-sum differential games. *Automatica*, 47(1), 207–214.

Zhang, H., Wang, Z., Liu, D. (2014). A Comprehensive Review of Stability Analysis of Continuous-Time Recurrent Neural Networks. *IEEE Trans on Neural Networks and Learning Systems*, 25(7), 1229–1262.

Zhang, H., Qing, C., Jiang, B., Luo, Y. (2014). Online Adaptive Policy Learning Algorithm for H_∞ State Feedback Control of Unknown Affine Nonlinear Discrete-Time Systems, *IEEE Transactions on Cybernetics*, 44(12), 2706–2718.