Actor–Critic-Based Optimal Tracking for Partially Unknown Nonlinear Discrete-Time Systems

Bahare Kiumarsi, *Student Member, IEEE*, and Frank L. Lewis, *Fellow, IEEE*

***Abstract*— This paper presents a partially model-free adaptive optimal control solution to the deterministic nonlinear discrete-time (DT) tracking control problem in the presence of input constraints. The tracking error dynamics and reference trajectory dynamics are first combined to form an augmented system. Then, a new discounted performance function based on the augmented system is presented for the optimal nonlinear tracking problem. In contrast to the standard solution, which finds the feedforward and feedback terms of the control input separately, the minimization of the proposed discounted performance function gives both feedback and feedforward parts of the control input simultaneously. This enables us to encode the input constraints into the optimization problem using a nonquadratic performance function. The DT tracking Bellman equation and tracking Hamilton–Jacobi–Bellman (HJB) are derived. An actor–critic-based reinforcement learning algorithm is used to learn the solution to the tracking HJB equation online without requiring knowledge of the system drift dynamics. That is, two neural networks (NNs), namely, actor NN and critic NN, are tuned online and simultaneously to generate the optimal bounded control policy. A simulation example is given to show the effectiveness of the proposed method.**

***Index Terms*— Actor–critic algorithm, discrete-time (DT) nonlinear optimal tracking, input constraints, neural network (NN), reinforcement learning (RL).**

I. Introduction

OPTIMAL control of nonlinear systems is an important topic in control engineering. Although most nonlinear control design methods concern only about the stability of the controlled systems, the stability is a bare minimum requirement and it is desired to design a controller by optimizing some predefined performance criteria. Optimal control problems can be divided into two major groups: optimal regulation and optimal tracking. The aim of the optimal regulation is to make the states of the system go to zero in an optimal manner and the aim of the optimal tracking is to make the states of the system track a reference trajectory.

Manuscript received March 25, 2014; revised September 3, 2014; accepted September 8, 2014. Date of publication October 8, 2014; date of current version December 16, 2014. This work was supported in part by the National Science Foundation under Grant ECCS-1405173 and Grant IIS-1208623, in part by the U.S. Office of Naval Research, Arlington, VA, USA, under Grant N00014-13-1-0562, in part by the Air Force Office of Scientific Research, Arlington, VA, USA, through the European Office of Aerospace Research and Development Project under Grant 13-3055, in part by the National Natural Science Foundation of China under Grant 61120106011, and in part by the 111 Project, Ministry of Education, China, under Grant B08015.

The authors are with the UTA Research Institute, University of Texas at Arlington, Fort Worth, TX 76118 USA (e-mail: kiumarsi@uta.edu; lewis@uta.edu).

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

Digital Object Identifier 10.1109/TNNLS.2014.2358227

It is well known that finding the optimal regulation solution requires solving the Hamilton–Jacobi–Bellman (HJB) equation, which is a nonlinear partial differential equation. Explicitly solving the HJB equation is generally very difficult or even impossible. Several techniques have been proposed to approximate the HJB solution. Included are reinforcement learning (RL) [1]–[8] and backpropagation through time [9]. RL techniques have been successfully applied to find the solution to the HJB equation online in real time for unknown or partially unknown continuous-time (CT) systems [10]–[12] and discrete-time (DT) systems [13]–[17]. Of the available RL schemes, the temporal difference learning method and approximate dynamic programming (ADP) [3], [18]–[26] have been used extensively to solve the optimal regulation problem because they do not require the knowledge of the system dynamics. The extensions of RL algorithms to solve multiplayer nonzero-sum games [27] and decentralized stabilization problem have been considered [28]. The optimal regulation control of nonlinear systems in the presence of input constraints has also been considered due to its importance in real world applications [29]–[31]. If the input constraints are not considered *a priori*, the control input may exceed its permitted bound because of the actuator saturation and this leads to performance degradation or even system instability.

In contrast to the solution to the optimal regulation problem, which consists of only a feedback term obtained by solving an HJB equation, the solution to the optimal tracking problems consists of two terms: a feedforward term that guarantees perfect tracking and a feedback term that stabilizes the system dynamics. Existing solutions to the nonlinear optimal tracking problems find the feedforward term using the dynamics inversion concept and the feedback term by solving an HJB equation [32]–[36]. To our knowledge, the problem of optimal tracking control of deterministic nonlinear DT systems with input constraints has not been considered yet. This is because the feedback and feedforward control terms are obtained separately and only the feedback term of the control input appears in the performance function, it is not possible to encode input constraints into the performance function, as it was done for the optimal regulation problem [29]–[31]. In addition, in the standard formulation of the optimal tracking problem, even if RL is used to find the feedback control term without any knowledge of the system dynamics, finding the feedforward term requires complete knowledge of the system dynamics. Recently, in [37] and [38], we presented ADP algorithms for solving the linear quadratic tracking problem without requiring complete knowledge of the system dynamics. In [37],

the Q-learning algorithm is presented to solve the optimal tracking problem of linear DT systems without any knowledge about the system dynamic or the reference trajectory dynamics. In [38], we solved optimal tracking problem for linear DT systems online without knowing the drift system dynamics or the reference trajectory using one-layer actor–critic structure. In [37] and [38], the value function and the control input are updated using an iterative algorithm. That is, once the value function is updated, the control policy is fixed and vice versa. In this paper, as an extension of [37] and [38], the optimal tracking problem for deterministic nonlinear DT systems by considering control input constraints is solved without knowing the drift system dynamics or the reference trajectory dynamics. The optimal tracking problem for nonlinear systems with input constraints is transformed into minimizing a nonqudratic performance function subject to an augmented system composed of the original system and the command generator system. It is shown that solving this problem requires solving a nonlinear tracking HJB equation. This is in contrast to [37] and [38] that require solving an algebraic Riccati equation.

The contributions of this paper are as follows.

1) A new formulation for the optimal tracking problem of deterministic nonlinear DT systems with input constraints is presented, which gives both feedback and feedforward parts of the control input simultaneously by solving an augmented tracking HJB equation.
2) A rigorous proof of stability and optimality of the solution to the tracking HJB equation is provided.
3) The proposed method satisfies the limitations of the amplitude of the control inputs imposed by physical limitations of the actuators.
4) An actor–critic structure with two-layer perceptron neural networks (NNs) is used to learn the solution to the tracking HJB equation. A new synchronous update law for the actor–critic structure is presented to obviate the requirement of the knowledge of the value function for the next time step.
5) The proposed method does not require the knowledge of the system drift dynamics or the command generator dynamics.

The rest of this paper is organized as follows. The next section formulates the optimal tracking control problem and discusses its standard solution and its shortcomings. In Section III, the new formulation to the optimal tracking problem of deterministic nonlinear DT constrained-input systems is presented. In addition, a DT tracking HJB equation is obtained, which gives both feedback and feedforward parts of the control input simultaneously. An actor–critic-based controller is given in Section IV, which learns the solution to the DT tracking HJB online and without requiring knowledge of the drift system dynamics or the reference trajectory dynamics. Finally, the simulation results are presented in Section V to confirm the suitability of the proposed method.

II. Problem Formulation and Its Standard Solution

In this section, we formulate the nonlinear optimal tracking problem and give the standard solution. The standard solution has several deficiencies. We fix these in the following sections, where we give our results.

Consider the deterministic nonlinear affine DT system given by

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

where $k > 0$, $x(k) \in \mathbb{R}^n$ represents the state vector of the system, $u(k) \in \mathbb{R}^m$ represents the control vector, $f(x(k)) \in \mathbb{R}^n$ is the drift dynamics of the system, and $g(x(k)) \in \mathbb{R}^{n\times m}$ is the input dynamics of the system. Assume that $f(0) = 0$ and $f(x) + g(x)u$ is Lipschitz continuous on a compact set Ω which contains the origin and the system (1) is controllable in the sense that there exists a continuous control on Ω which stabilizes the system.

For the infinite-horizon tracking problem, the goal is to design an optimal controller for the system (1) which ensures that the state $x(k)$ tracks the reference trajectory $r(k)$ in an optimal manner. Define the tracking error as

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

For the standard solution to the tracking problem, the control input consists of two parts, including a feedforward part and a feedback part [5]. In the following, it is discussed how each of these parts are obtained using the standard method.

The feedforward or steady-state part of the control input is used to assure perfect tracking. The perfect tracking is achieved if $x(k) = r(k)$. In order for this to happen, a feedforward control input $u(k) = u_d(k)$ must exist to make $x(k)$ equal to $r(k)$. By substituting $x(k) = r(k)$ and $u(k) = u_d(k)$ in the system dynamics (1), one has

$$r(k+1) = f(r(k)) + g(r(k))\,u_d(k). \tag{3}$$

If the system dynamics is known, $u_d(k)$ is obtained by

$$u_d(k) = g(r(k))^+ \left(r(k+1) - f(r(k))\right) \tag{4}$$

where $g(r(k))^+ = (g(r(k))^T g(r(k)))^{-1} g(r(k))^T$ is the generalized inverse of $g(r(k))$ with $g(r(k))^+ g(r(k)) = I$.

Remark 1: Note that (4) cannot be solved for $u_d(k)$ unless the reference trajectory is given *a priori*. Moreover, finding $u_d(k)$ requires the inverse of the input dynamics $g(r(k))$ and the function $f(r(k))$ be known. We shall propose a new method in Section III that does not need the reference trajectory to be given or the input dynamics to be invertible.

The feedback control input is computed as follows. Using (1) and (3), the tracking error dynamics can be expressed in terms of the tracking error $e(k)$ and the reference trajectory $r(k)$ as

$$\begin{aligned} e(k+1) &= x(k+1) - r(k+1) \\ &= f(e(k)+r(k)) + g(e(k)+r(k))u_e(k) \\ &\quad + g(e(k)+r(k))g(r(k))^+(r(k+1) - f(r(k))) \\ &\quad - r(k+1). \end{aligned} \tag{5}$$

By defining

$$\begin{aligned} g_e(k) &\triangleq g(e(k)+r(k)) \\ f_e(k) &\triangleq f(e(k)+r(k)) + g(e(k)+r(k))g(r(k))^+ \\ &\quad (r(k+1) - f(r(k))) - r(k+1). \end{aligned} \tag{6}$$

Equation (5) can be rewritten as

$$e(k+1) = f_e(k) + g_e(k)u_e(k) \tag{7}$$

where $u_e(k)$ is the feedback control input that is designed to stabilize the tracking error dynamics in an optimal manner by minimizing the following performance function:

$$J(e(k), u_e(k)) = \sum_{i=k}^{\infty} \left[e^T(i)\, Q_e\, e(i) + u_e^T(i)\, R_e\, u_e(i) \right] \tag{8}$$

where $Q_e \in \mathbb{R}^{n\times n}$ and $R_e \in \mathbb{R}^{m\times m}$ are symmetric positive definite. The feedback input that minimizes (8) is found by solving the stationarity condition $\partial J(e, u_e)/\partial u_e = 0$ [5]. The result is

$$u_e^*(k) = -\frac{1}{2} R_e^{-1} g_e^T(k) \frac{\partial J(e(k+1))}{\partial e(k+1)}. \tag{9}$$

Then, the standard control input is given by

$$u^*(k) = u_d(k) + u_e^*(k) \tag{10}$$

where u_d is given by (4) and u_e^* is given by (9).

Remark 2: The feedback part of the control input (9) is designed to stabilize the tracking error dynamics. The RL algorithm for finding the feedback part could be implemented by measuring the state $x(k)$ and the reference trajectory $r(k)$ to obtain $e(k)$ and without requiring the system dynamics [5]. In contrast, obtaining the feedforward part of the control input needs complete knowledge of the system dynamics and the reference trajectory dynamics.

Remark 3: Since the feedforward part of the control input (4) is found separately and does not involve in the optimization of the performance (8), it is not possible to encode the constraints of the control input u into the optimization problem. In the following, a new formulation of the problem is given that gives both feedback and feedforward parts of the control input simultaneously by minimizing a predefined performance function.

III. New Formulation for the Nonlinear Input Constraints Tracking Problem

A disadvantage to the standard tracking problem solution in Section II is that it needs complete knowledge of the system dynamics, and also the reference trajectory should satisfy (3) to compute the feedforward control input (4). Moreover, the standard solution does not consider the input constraints caused by physical limitations of the actuator. In this section, we propose an alternative approach for formulating the nonlinear tracking problem that gives both parts of the control input simultaneously and also considers the input constraints. First, an augmented system composed of the original system and the reference trajectory is constructed. Based on this augmented system, a new performance function is presented that consists of both feedback and feedforward parts of the control input. Then, the Bellman and HJB equations for the nonlinear tracking problem are obtained.

A. Augmented System Dynamics and Discounted Performance Function

Before proceeding, the following assumption is made for the reference trajectory.

Assumption 1: The reference trajectory dynamics for the nonlinear tracking problem is generated by the command generator model

$$r(k+1) = \psi(r(k)) \tag{11}$$

where $\psi(r(k))$ is a C^∞ function with $\psi(0) = 0$ and $r(k) \in \mathbb{R}^n$ [39].

Definition 1: A equilibrium point is said to be Lyapunov stable if for every $\varepsilon > 0$, there exists a $\delta = \delta(\varepsilon) > 0$ such that, if $\|x(0) - x_e\| < \delta$, then for every $t \geq 0$ we have $\|x(t) - x_e\| < \varepsilon$.

Remark 4: Note that Assumption 1 is a standard assumption made in accordance with other work on tracking control in [39]. This command generator dynamics can generate a large class of command trajectories, including unit step, sinusoidal waveforms, and damped sinusoids. Human factor studies show that after learning a task, the skilled human operator behaves predictably like a command generator dynamics. Assume, the model (11) covers this case from human–robot interaction.

The tracking error dynamics (5) in terms of the control input $u(k)$ is given by

$$\begin{aligned} e(k+1) = x(k+1) - r(k+1) &= f(e(k) + r(k)) \\ &\quad - \psi(r(k)) + g(e(k) + r(k))u(k). \end{aligned} \tag{12}$$

Based on (11) and (12), the augmented system is constructed in terms of the tracking error $e(k)$ and the reference trajectory $r(k)$ as

$$\begin{aligned} \begin{bmatrix} e(k+1) \\ r(k+1) \end{bmatrix} &= \begin{bmatrix} f(e(k) + r(k)) - \psi(r(k)) \\ \psi(r(k)) \end{bmatrix} \\ &\quad + \begin{bmatrix} g(e(k) + r(k)) \\ 0 \end{bmatrix} u(k) \\ &\equiv F(X(k)) + G(X(k))u(k) \end{aligned} \tag{13}$$

where the augmented state is

$$X(k) = \begin{bmatrix} e(k) \\ r(k) \end{bmatrix} \in \mathbb{R}^{2n} \tag{14}$$

and

$$u(k) \in \mathbb{R}^m \,\big|\, \; |u_j(k)| \leq \bar{u}_j, \quad j = 1, \ldots, m \tag{15}$$

where $\bar{u}_j$ is the saturating bound for the jth actuator. Based on the augmented system (13), define the value function for the tracking problem as

$$\begin{aligned} V(X(k)) &= \sum_{i=k}^{\infty} \gamma^{i-k} U_i \\ &= \sum_{i=k}^{\infty} \gamma^{i-k} \left[X(i)^T Q_1 X(i) + W(u(i)) \right] \end{aligned} \tag{16}$$

where

$$Q_1 = \begin{bmatrix} Q & 0 \\ 0 & 0 \end{bmatrix}. \tag{17}$$

Here, $0 < \gamma \leq 1$ is the discount factor and $W(u(i)) \in \mathbb{R}$ is used to penalize the control input. Q and $W(u(i))$ are positive definite.

For the unconstrained control problem, $W(u)$ may have the quadratic form $W(u(i)) = u(i)^T R\, u(i)$. However, to deal with the constrained control input [29]–[31], [40], [41] we employ a nonquadratic functional defined as

$$W(u(i)) = 2\int_0^{u(i)} \varphi^{-T}(\bar{U}^{-1}s)\,\bar{U}\,R\,ds \tag{18}$$

where $\bar{U} \in \mathbb{R}^{m\times m}$ is the constant diagonal matrix given by $\bar{U} = \text{diag}\{\bar{u}_1, \bar{u}_2, \ldots, \bar{u}_m\}$ and R is positive definite and assumed to be diagonal. $\varphi(.)$ is a bounded one-to-one function satisfying $|\varphi(.)| < 1$. Moreover, it is a monotonic increasing odd function with its first derivative bounded by a constant L. It is clear that $W(u(i))$ in (18) is a scalar for $u(i) \in \mathbb{R}^m$. $W(u)$ is ensured to be positive definite because φ is monotonic odd function and R is positive definite. This function is written in terms of scalar components as follows:

$$\varphi^{-1}(\bar{U}^{-1}s) = \left[\varphi^{-1}\left(\tfrac{u_1(i)}{\bar{u}_1(i)}\right) \quad \varphi^{-1}\left(\tfrac{u_2(i)}{\bar{u}_2(i)}\right) \quad \ldots \quad \varphi^{-1}\left(\tfrac{u_m(i)}{\bar{u}_m(i)}\right)\right]^T \tag{19}$$

where $s \in \mathbb{R}^m$. Denote $\omega(s) = \varphi^{-T}(\bar{U}^{-1}s)\bar{U}R = [\omega_1(s_1)\ldots\omega_m(s_m)]$. Then, the integral in (18) is defined in terms of the components of $u(i)$ as [42]

$$W(u(i)) = 2\int_0^{u(i)} \omega(s)\,ds = 2\sum_{j=1}^{m}\int_0^{u_j(i)} \omega_j(s_j)\,ds_j. \tag{20}$$

Note that by defining the nonquadratic function $W(u(i))$ (18), the control input resulting by minimizing the value function (16) is bounded (29). The function (16) has been used to generate bounded control for CT systems in [29]–[31].

Remark 5: Note that it is essential to use a discounted performance function for the proposed formulation. This is because if the reference trajectory does not go to zero, which is the case of most real applications, then the performance function (16) becomes infinite without the discount factor as the control input contains a feedforward part, which depends on the reference trajectory, and thus $W(u)$ does not go to zero as time goes to infinity.

B. Bellman and HJB Equations for the Nonlinear Tracking Problem

In this section, based on the augmented system and the value function presented in the previous section, the Bellman and HJB equations for the nonlinear tracking problem are given.

Using (16) and (18), we have

$$V(X(k)) = X(k)^T Q_1\, X(k) + 2\int_0^{u(k)} \varphi^{-T}(\bar{U}^{-1}s)\,\bar{U}\,R\,ds + \sum_{i=k+1}^{\infty} \gamma^{i-k}\left[X(i)^T Q_1\, X(i) + 2\int_0^{u(i)} \varphi^{-T}(\bar{U}^{-1}s)\,\bar{U}\,R\,ds\right] \tag{21}$$

which yields the nonlinear tracking Bellman equation

$$V(X(k)) = X(k)^T Q_1\, X(k) + 2\int_0^{u(k)} \varphi^{-T}(\bar{U}^{-1}s)\,\bar{U}\,R\,ds + \gamma\, V(X(k+1)). \tag{22}$$

Based on (22), define the Hamiltonian function

$$H(X(k), u(k), V) = X(k)^T Q_1\, X(k) + 2\int_0^{u(k)} \varphi^{-T}(\bar{U}^{-1}s)\bar{U}R\,ds + \gamma\, V(X(k+1)) - V(X(k)). \tag{23}$$

From Bellman's optimality principle, it is well known that, for the infinite-horizon optimization case, the value function $V^*(X(k))$ is time-invariant and satisfies the DT HJB equation

$$V^*(X(k)) = \min_{u(k)}\left[X(k)^T Q_1\, X(k) + 2\int_0^{u(k)} \varphi^{-T}(\bar{U}^{-1}s)\,\bar{U}R\,ds + \gamma\, V^*(X(k+1))\right]. \tag{24}$$

Or equivalently

$$H(X(k), u^*(k), V^*) = X(k)^T Q_1\, X(k) + 2\int_0^{u^*(k)} \varphi^{-T}(\bar{U}^{-1}s)\,\bar{U}\,R\,ds + \gamma\, V^*(X(k+1)) - V^*(X(k)). \tag{25}$$

Necessary condition for optimality is stationarity condition [5]

$$\begin{aligned}\frac{\partial H(X(k), u(k), V^*)}{\partial u(k)} &= \frac{\partial\left(X(k)^T Q_1\, X(k) + 2\int_0^{u(k)} \varphi^{-T}(\bar{U}^{-1}s)\,\bar{U}\,R\,ds\right)}{\partial u(k)} + \gamma\left(\frac{\partial X(k+1)}{\partial u(k)}\right)^T \frac{\partial V^*(X(k+1))}{\partial X(k+1)} \\ &= 0.\end{aligned} \tag{26}$$

Using (13), we have

$$\frac{\partial X(k+1)}{\partial u(k)} = G(X) \tag{27}$$

also

$$\frac{\partial\left(X(k)^T Q_1\, X(k) + 2\int_0^{u(k)} \varphi^{-T}(\bar{U}^{-1}s)\,\bar{U}\,R\,ds\right)}{\partial u(k)} = 2\varphi^{-T}(\bar{U}^{-1}u(k))\,\bar{U}\,R. \tag{28}$$

By substituting (27) and (28) into (26), yields the optimal control input

$$u^*(k) = \bar{U}\varphi\left(-\frac{\gamma}{2}(\bar{U}R)^{-T} G(X)^T \frac{\partial V^*(X(k+1))}{\partial X(k+1)}\right). \tag{29}$$

By substituting (29) in the Bellman equation, the DT tracking HJB equation (24) becomes

$$V^*(X(k)) = X(k)^T Q_1 X(k) + W(u^*(k)) + \gamma V^*(X(k+1)) \tag{30}$$

where the $V^*(X(k))$ is the value function corresponding to the optimal control input.

To find the optimal control solution, first the DT tracking HJB equation (30) is solved and then the optimal control solution is given by (29) using the solution of the DT tracking HJB equation. However, in general, the DT tracking HJB equation cannot be solved analytically. In the subsequent sections, it is shown how to solve the DT tracking HJB equation online in real-time without complete knowledge of the augmented system dynamics.

Remark 6: Note that in (29) both feedback and feedforward parts of the control input are obtained simultaneously by minimizing the performance function (16) subject to the augmented system (13). This enables us to extend RL techniques for solving the nonlinear optimal tracking problem without using complete knowledge of the system dynamics. In addition, it enables us to encode the input constraints into the optimization problem using the nonquadratic function (18).

Remark 7: It is clear from (29) that the optimal control input never exceeds its permitted bounds. This is a result of reformulation of the nonlinear tracking problem to minimizing of the nonquadratic performance function (16) subject to the augmented system (13). In this formulation, both feedback and feedforward parts of the control input are obtained simultaneously by minimizing the performance function (16) and the control input stays in its permitted bounds because of using the nonquadratic function (18) for penalizing the control input.

C. Optimal Tracking Problem With Input Constraints

The following Theorem 1 shows that the solution obtained by the DT tracking HJB equation gives the optimal solution and locally asymptotically stabilizes the error dynamics (12) in the limit as the discount factor goes to one.

Lemma 1: For any admissible control policy $u(k)$, suppose $V(X(k)) \in C^1 : \mathbb{R}^n \to \mathbb{R}$ is a smooth positive definite solution to the Bellman equation (22). Define $u^* = u(V(X))$ by (29) in terms of $V(X)$. Then

$$\begin{aligned} &H(X(k), u(k), V) \\ &\quad = H(X(k), u^*(k), V) + 2\int_{u^*(k)}^{u(k)} \varphi^{-T}(\bar{U}^{-1}s)\,\bar{U}\,R\,ds \\ &\qquad + \gamma V(F(X(k)) + G(X(k))u) \\ &\qquad - \gamma V(F(X(k)) + G(X(k))u^*). \end{aligned} \tag{31}$$

Proof: The Hamiltonian function is

$$\begin{aligned} &H(X(k), u(k), V) \\ &= X(k)^T Q_1 X(k) + 2\int_0^{u(k)} \varphi^{-T}(\bar{U}^{-1}s)\,\bar{U}\,R\,ds \\ &\quad + \gamma V(X(k+1)) - V(X(k)). \end{aligned} \tag{32}$$

By adding and subtracting the terms $2\int_0^{u^*(k)} \varphi^{-T}(\bar{U}^{-1}s)\,\bar{U}\,R\,ds$ and $\gamma V(F(X(k)) + G(X(k))u^*)$ to (32) yields

$$\begin{aligned} &H(X(k), u(k), V) \\ &\quad = X(k)^T Q_1 X(k) + 2\int_0^{u^*(k)} \varphi^{-T}(\bar{U}^{-1}s)\,\bar{U}\,R\,ds \\ &\qquad + \gamma V(F(X(k)) + G(X(k))u^*) - V(X(k)) \\ &\qquad + 2\int_{u^*(k)}^{u(k)} \varphi^{-T}(\bar{U}^{-1}s)\,\bar{U}\,R\,ds + \gamma V(F(X(k)) \\ &\qquad + G(X(k))u) - \gamma V(F(X(k)) + G(X(k))u^*). \end{aligned} \tag{33}$$

The proof is complete. □

Theorem 1 (Solution to the Optimal Control Problem): Consider the augmented system (13) with performance function (16). Suppose $V^*(X(k)) \in C^1 : \mathbb{R}^n \to \mathbb{R}$ is a smooth positive definite solution to the DT HJB equation (30). Define control $u^* = u(V^*(k))$ as given by (29). Then, the closed-loop system

$$\begin{aligned} X(k+1) &= F(X(k)) + G(X(k))u^* \\ &= F(X(k)) + G(X(k))\,\bar{U} \\ &\quad \times \varphi\left(-\frac{\gamma}{2}(\bar{U}R)^{-T} G(X)^T \frac{\partial V^*(X(k+1))}{\partial (X(k+1))}\right) \end{aligned} \tag{34}$$

is locally asymptotically stable in the limit, as the discount factor goes to one. Moreover, $u^* = u(V^*(k))$ is bounded and minimizes the value function (16) over all bounded controls, and the optimal value on $[0, \infty)$ is given by $V^*(X(k))$.

Proof: Clearly, (29) is bounded**.** The proof of the stability and optimality of the DT tracking HJB equation is given separately as follows.

1) Stability: Suppose that the value function $V^*(X(k))$ satisfies the DT tracking HJB equation. Then, using (30) one has

$$V^*(X(k)) - \gamma V^*(X(k+1)) = X(k)^T Q_1 X(k) + W(u^*(k)). \tag{35}$$

Multiplying both sides of (35) by γ^k gives

$$\begin{aligned} &\gamma^k V^*(X(k)) - \gamma^{k+1} V^*(X(k+1)) \\ &\qquad = \gamma^k \big(X(k)^T Q_1 X(k) + W(u^*(k))\big). \end{aligned} \tag{36}$$

Defining the difference of the Lyapunov function as $\Delta(\gamma^k V^*(X(k))) = \gamma^{(k+1)} V^*(X(k+1)) - \gamma^k V^*(X(k))$ and using (36) yields

$$\Delta(\gamma^k V^*(X(k))) = -\gamma^k (X(k)^T Q_1 X(k) + W(u^*(k))). \tag{37}$$

Equation (37) shows that the tracking error is bounded for the optimal solution, but its asymptotic stability cannot be concluded. However, if $\gamma = 1$ (which can be chosen only if the reference trajectory goes to zero), LaSalle's extension [43, p. 113] can be used to show that the tracking error is locally asymptotically stable. The LaSalle's extension says that the states of the system converge to a region wherein $\dot{V} = 0$. Using (37), we have $\dot{V} = 0$ if $e(k) = 0$ and $W(u(k)) = 0$. Since $W(u(k)) = 0$ if $e(k) = 0$, therefore for $\gamma = 1$, the tracking error is locally asymptotically stable. This confirms that in the limit as the discount factor goes to one, the optimal control input resulted from solving the DT

tracing HJB equation makes the error dynamics asymptotically stable.

If the discount factor is chosen as a nonzero value, one can make the tracking error as small as desired by choosing a discount factor close to one in the performance function (16).

2) Optimality: We now prove that the HJB equation provides the sufficient condition for optimality. Note that for any admissible control input $u(k)$ and initial condition $X(0)$, one can write the value function (16) as

$$\begin{aligned} V(X(0),u) &= \sum_{i=0}^{\infty} \gamma^i [X(i)^T Q_1 X(i) + W(u(i))] \\ &= \sum_{i=0}^{\infty} \gamma^i [X(i)^T Q_1 X(i) + W(u(i))] \\ &\quad + \sum_{i=0}^{\infty} \gamma^i [\gamma V(X(k+1)) - V(X(k))] + V(X(0)). \end{aligned} \tag{38}$$

Then

$$V(X(0),u) = \sum_{i=0}^{\infty} \gamma^i H(X,u,V) + V(X(0)). \tag{39}$$

By substituting (31) into (39) and considering $H(X^*,u^*,V^*) = 0$ and (38) yields

$$\begin{aligned} &V(X(0),u) \\ &= \sum_{i=0}^{\infty} \gamma^i \left[2 \int_{u^*(k)}^{u(k)} \varphi^{-T}(\bar{U}^{-1}s)\, \bar{U}\, R\, ds + \gamma V^*(F(X(k)) \right. \\ &\qquad \left. + G(X(k))u) - \gamma V^*(F(X(k)) + G(X(k))u^*) \right] \\ &\quad + V^*(X(0)). \end{aligned} \tag{40}$$

Or equivalent

$$V(X(0),u) = M + V^*(X(0)) \tag{41}$$

where

$$\begin{aligned} M = \sum_{i=0}^{\infty} \gamma^i \left[2 \int_{u^*(k)}^{u(k)} \varphi^{-T}(\bar{U}^{-1}s)\, \bar{U}\, R\, ds + \gamma V^*(F(X(k)) \right. \\ \left. + G(X(k))u) - \gamma V^*(F(X(k)) + G(X(k))\, u^*) \right]. \end{aligned} \tag{42}$$

To show that u^* is an optimal control and the optimal value function is $V^*(X(0))$, it is needed to show that the term M is bigger than zero when $u \neq u^*$ and it is zero when $u = u^*$.

Taking $\varphi^{-1}(.)$ from two sides of (29), yields

$$2(\bar{U}R)^T \varphi^{-1}(\bar{U}^{-1}u^*) = -\gamma\, G(X)^T \frac{\partial V^*(X(k+1))}{\partial X(k+1)}. \tag{43}$$

In addition, we have

$$\begin{aligned} \frac{dV^*(X(k+1))}{du(k)} &= \left(\frac{\partial V^*(X(k+1))}{\partial X(k+1)} \right)^T \frac{\partial X(k+1)}{\partial u(k)} \\ &= \left(\frac{\partial V^*(X(k+1))}{\partial X(k+1)} \right)^T G(X). \end{aligned} \tag{44}$$

Using (44) in (43), yields

$$2(\bar{U}R)^T \varphi^{-1}(\bar{U}^{-1}u^*) = -\gamma \left(\frac{dV^*(X(k+1))}{du(k)} \right)^T. \tag{45}$$

By integrating (45) one has

$$\int_{u^*}^{u} 2\, \varphi^{-T}(\bar{U}^{-1}u^*)\, \bar{U}R\, ds = -\gamma \int_{u^*}^{u} \frac{dV^*(X(k+1))}{ds(k)}\, ds$$

Or equivalently

$$\begin{aligned} 2(u-u^*)\, \varphi^{-T}(\bar{U}^{-1}u^*)\bar{U}R = -\gamma [V^*(F(X(k))+G(X(k))u) \\ -V^*(F(X(k))+G(X(k))u^*)]. \end{aligned} \tag{46}$$

Using (46) and considering $\alpha(s) = \varphi^{-T}(\bar{U}^{-1}s)\bar{U}\, R$, M becomes

$$M = \sum_{i=0}^{\infty} \gamma^i \left[2 \int_{u^*(k)}^{u(k)} \alpha(s)\, ds - 2(u-u^*)\, \alpha(u^*) \right]. \tag{47}$$

We consider

$$l = \int_{u^*(k)}^{u(k)} \alpha(s) ds - (u-u^*)\, \alpha(u^*). \tag{48}$$

Using (20), we rewrite (48) as

$$l = \sum_{j=1}^{m} \int_{u_j^*(k)}^{u_j(k)} \alpha_j(s_j) ds_j - (u_j - u_j^*)\, \alpha_j(u_j^*). \tag{49}$$

By defining $l_j = \int_{u_j^*(k)}^{u_j(k)} \alpha_j(s_j) ds_j - (u_j - u_j^*)\, \alpha_j(u_j^*)$, (49) is

$$l = \sum_{j=1}^{m} l_j. \tag{50}$$

To complete the proof, it is needed to show that l_j, $j = 1, \ldots, m$ is bigger than zero for $u_j \neq u_j^*$ and they are zero for $u_j = u_j^*$. It is clear that $l_j = 0$ for $u_j = u_j^*$. Now, it should be shown that $l_j > 0$ for $u_j \neq u_j^*$. First, it is assumed $u_j > u_j^*$. Then, using mean value theorem for the integrals, there exists a $\bar{u}_j \in [u_j^*, u_j]$ such that

$$\int_{u_j^*(k)}^{u_j(k)} \alpha_j(s_j) ds_j = (u_j - u_j^*)\, \alpha(\bar{u}_j) > (u_j - u_j^*)\, \alpha_j(u_j^*). \tag{51}$$

Therefore, $l_j > 0$ for $u_j > u_j^*$. Then, it is assumed $u_j < u_j^*$. Using mean value theorem, there exists a $\bar{u}_j \in [u_j, u_j^*]$ such that

$$\int_{u_j^*(k)}^{u_j(k)} \alpha_j(s_j) ds_j = (u_j - u_j^*)\, \alpha(\bar{u}_j) > (u_j - u_j^*)\, \alpha_j(u_j^*). \tag{52}$$

Therefore, $l_j > 0$ for $u_j < u_j^*$. Then, the proof is completed. □

IV. RL for Solving the Nonlinear Tracking Problem

The DT HJB equation (30) cannot be solved exactly and it also requires complete knowledge of the system dynamics. In this section, an online policy iteration algorithm is given to find the solution to the DT tracking HJB equation. This algorithm still requires complete knowledge of the system dynamics. To obviate the requirement of complete knowledge of the system dynamics, an actor–critic algorithm is presented in the following section to solve the nonlinear tracking problem.

Note that instead of the solving the DT HJB equation directly, one can use the following iterative policy iteration algorithm which uses the tracking Bellman equation (22) to evaluate a fixed control policy and the update policy in form of (29) to find an improved control policy.

A. Algorithm 1. Online Policy Iteration Algorithm

Initialize the control input $u^0(k)$.

1) Policy Evaluation: Find the value function related to the policy $u^i(k)$ by solving the Bellman equation (22)

$$V^i(X(k)) = X(k)^T Q_1 X(k) + W(u^i(k)) + g V^i(X(k+1)). \quad (53)$$

2) Policy Improvement: Update the policy using

$$u^{i+1}(X) = \bar{U}\varphi\left(-\frac{\gamma}{2}(\bar{U}R)^{-T} G(X(k))^T \frac{\partial V^i(X(k+1))}{\partial X(k+1)}\right). \quad (54)$$

Remark 8: This policy iteration algorithm can be implemented online using the least squares method by standard technique [6]. This requires a persistence of excitation (PE) condition to allow solution of repeated Bellman equation (53) at successive time instants in a batch fashion.

The augmented system dynamics is not needed to solve the Bellman equation (53) but must be known for updating the control input using (54). To obviate the requirement complete knowledge of the system dynamics or reference trajectory dynamics, an actor–critic algorithm [6] is developed in the following section to solve the nonlinear tracking problem.

B. Actor–Critic Structure for Solving the Nonlinear Tracking Problem

In this section, the solution to the DT tracking HJB equation is learned using an actor–critic structure which does not require knowledge of drift system dynamics or reference trajectory dynamics. In contrast to the sequential online Algorithm 1, the proposed algorithm is an online synchronized algorithm. That is, instead of sequentially updating the value function and the policy, as shown in Algorithm 1, the value function and the policy are updated simultaneously. This method has been developed for CT systems in [44] and for DT systems with one-layer NN in [16]. The value function and the control input are approximated with two separate two-layer perceptron NNs [18]–[20]. The critic NN estimates the value function and is a function of the tracking error and the reference trajectory. The actor NN represents a control policy and is a function of the tracking error and the reference trajectory. The critic NN is updated to minimize the tracking Bellman error and the actor NN is to minimize the value function.

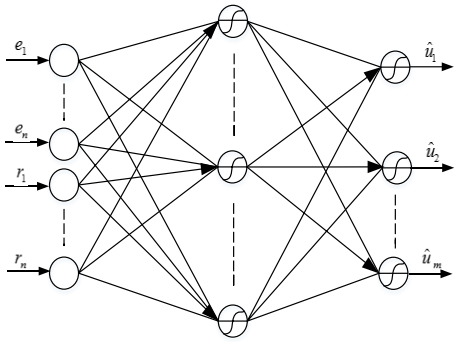

Fig. 1. Actor network.

1) Actor NN: A two-layer NN with one hidden layer is used as the actor NN to approximate the control input for the nonlinear tracking problem. Fig. 1 shows the actor NN. The input for the actor NN is $X(k)$ defined in (14).

The output of the actor NN are given as

$$\begin{aligned}\hat{u}(X(k), W_a^{(1)}(k), W_a^{(2)}(k)) &= [\hat{u}_1, \ldots, \hat{u}_m] \\ \hat{u}_i &= \sigma_i(W_{a_i}^{(2)}(k)\,\phi_a(W_a^{(1)}(k)X(k))) \\ &= \sigma_i\left(\sum_{l=1}^{N_a} \hat{w}_{a_{il}}^{(2)}(k)\,\phi_{a_l}\left(\sum_{j=1}^{2n} \hat{w}_{a_{lj}}^{(1)}(k)X_j(k)\right)\right) \quad (55)\end{aligned}$$

where $\phi_a(k) = [\phi_{a_1}(k), \ldots, \phi_{a_{N_a}}(k)]^T \in \mathbb{R}^{N_a \times 1}$ is the vector of hidden-layer activation functions with $\phi(.) = \tanh(.)$, and $\sigma_i(.) = \bar{u}_i \tanh(.)$, $i = 1, \ldots, m$ is the ith output-layer activation function.

The weight matrix between the neurons in the input and the hidden layers is defined as

$$W_a^{(1)} = \begin{bmatrix} w_{a_{11}}^{(1)} & \ldots & w_{a_{1(2n)}}^{(1)} \\ \vdots & \vdots & \vdots \\ w_{a_{N_a 1}}^{(1)} & \ldots & w_{a_{N_a(2n)}}^{(1)} \end{bmatrix} \in \mathbb{R}^{N_a \times 2n}$$

where $N_a$is number of the hidden-layer neurons and each element $w_{a_{ij}}^{(1)}$ denotes the weight from jth input to ith hidden neurons. In addition, define the matrix of the weights between the neurons in the hidden and the output layers as

$$W_a^{(2)} = \begin{bmatrix} w_{a_{11}}^{(2)} & \ldots & w_{a_{1N_a}}^{(2)} \\ \vdots & \vdots & \vdots \\ w_{a_{m1}}^{(2)} & \ldots & w_{a_{mN_a}}^{(2)} \end{bmatrix} \in \mathbb{R}^{m \times N_a}$$

where $\hat{w}_{a_{ij}}^{(2)}(k)$ is the weight between jth hidden layer and ith output layer.

Note that the output of the actor NN satisfies the input bounds defined in (15).

2) Critic NN: A two-layer NN with one hidden layer is used as critic NN to approximate the value function. Fig. 2 shows the critic NN. The input for critic NN is $X(k)$.

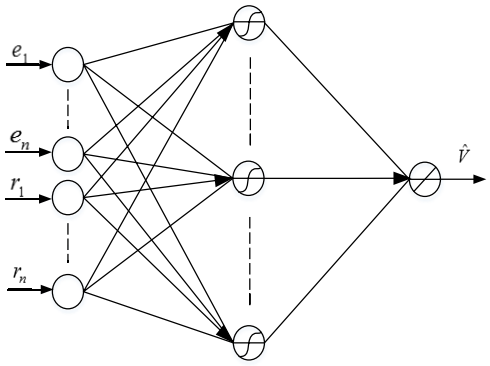

Fig. 2. Critic network.

The output of the critic NN is given as

$$\hat{V}(X(k)) = W_c^{(2)}(k)\,\phi_c(W_c^{(1)}(k)X(k)) = \sum_{i=1}^{N_c} \hat{w}_{c_i}^{(2)}(k)\,\phi_{c_i}\left(\sum_{j=1}^{2n} \hat{w}_{c_{ij}}^{(1)}(k)X_j(k)\right) \quad (56)$$

where $\phi_c(k) = [\phi_{c_1}(k), \ldots, \phi_{c_{N_c}}(k)]^T \in \mathbb{R}^{N_c \times 1}$ is the vector of hidden layer activation functions, with $\phi(.) = \tanh(.)$ and the matrix of the weights between the neurons in the input and the hidden layers is defined as

$$W_c^{(1)} = \begin{bmatrix} w_{c_{11}}^{(1)} & \ldots & w_{c_{1(2n)}}^{(1)} \\ \vdots & \vdots & \vdots \\ w_{c_{N_c 1}}^{(1)} & \ldots & w_{c_{N_c(2n)}}^{(1)} \end{bmatrix} \in \mathbb{R}^{N_c \times 2n}$$

where N_c is number of the hidden layer neurons and each element $w_{c_{ij}}^{(1)}$ denotes the weight from jth input to ith hidden neurons. In addition, define the vector of the weights between the neurons in the hidden and the output layer as

$$W_c^{(2)} = \left[w_{c_1}^{(2)}, w_{c_2}^{(2)}, ..., w_{c_{N_c}}^{(2)}\right] \in \mathbb{R}^{1 \times N_c}$$

where $\hat{w}_{c_i}^{(2)}(k)$ is the weight between ith hidden layer and output layer.

C. Learning Rules for Actor and Critic NNs

In this section, the approximate gradient descent rules for updating the critic and actor networks are developed. The objective of tuning the critic weights is to minimize the Bellman equation error and the objective of tuning the actor weights is to minimize the approximate value function. To obviate the need for $V(X(k+1))$, which requires the system model to predict $X(k+1)$, the previous values of the system state $X(k)$ and the state value $V(X(k))$ are stored and used for updating the actor and critic NNs weights.

1) Updating Rule for the Critic Network: The prediction error of the tracking Bellman equation is defined as

$$e_c(k) = \gamma\,\hat{V}(X(k)) + U_k - \hat{V}(X(k-1)) \quad (57)$$

where

$$U_k = X(k)^T Q_1\, X(k) + 2\int_0^{u(k)} \varphi^{-T}(\bar{U}^{-1}s)\,\bar{U}\,R\,ds$$

is the reinforcement signal or the temporal difference signal. It is desired to select the weights of the critic network to minimize the square Bellman error

$$E_c(k) = \frac{1}{2}e_c^2(k). \quad (58)$$

The tuning laws for the critic weights are selected as the normalized approximation to gradient descent algorithm, that is

$$\hat{w}_{c_{ij}}^{(1)}(k+1) = \hat{w}_{c_{ij}}^{(1)}(k) - l_c\frac{\partial E_c(k)}{\partial \hat{w}_{c_{ij}}^{(1)}(k)} \quad (59)$$

$$\hat{w}_{c_i}^{(2)}(k+1) = \hat{w}_{c_i}^{(2)}(k) - l_c\frac{\partial E_c(k)}{\partial \hat{w}_{c_i}^{(2)}(k)} \quad (60)$$

where $l_c > 0$ is learning rate.

Using the chain rule one has

$$\frac{\partial E_c(k)}{\partial \hat{w}_{c_{ij}}^{(1)}(k)} = \frac{\partial E_c(k)}{\partial e_c(k)}\frac{\partial e_c(k)}{\partial \hat{V}(k)}\frac{\partial \hat{V}(k)}{\partial \phi_{c_i}(k)}\frac{\partial \phi_{c_i}(k)}{\partial \hat{w}_{c_{ij}}^{(1)}(k)} = \gamma\; e_c(k)\hat{w}_{c_i}^{(2)}(k)\left(1-\phi_{c_i}^2(k)\right)X_j(k) \quad (61)$$

$$\frac{\partial E_c(k)}{\partial \hat{w}_{c_i}^{(2)}(k)} = \frac{\partial E_c(k)}{\partial \hat{V}(k)}\frac{\partial \hat{V}(k)}{\partial \hat{w}_{c_i}^{(2)}(k)} = \gamma\; e_c(k)\,\phi_{c_i}(k). \quad (62)$$

Note that there are no convergence guarantees with this kind of weight update since it is an approximation to gradient descent but it proved successful in the experiments in this paper. One can refer to [9], [45], and [46] for an exact gradient descent algorithm with improved convergence guarantees.

2) Updating Rule for the Actor Network: The actor (55) is updated to minimize the approximated value function. This is done by minimizing the error between a target control input that is obtained by minimizing the value function and the actual control input that is applied to the system. Let the current estimation of the value function be $\hat{V}$. Then, based on the policy update law (54), the target control input at time k is

$$\tilde{u}(X) = \bar{U}\varphi\left(-\frac{\gamma}{2}(\bar{U}R)^{-1}G(X(k))^T\frac{\partial \hat{V}(X(k+1))}{\partial X(k+1)}\right). \quad (63)$$

However, to obtain the value at time $k+1$, the states are required to be predicted using a model network. However, we do not use a model network to predict the future value. Rather, we store the previous value of the system state and the state value and try to minimize the error between the target control input and the actual control input given current actor and critic estimate weights, while the previous stored state $X(k-1)$ is used as the input to the actor and critic. That is to minimize

$$e_a(k) = \tilde{u}(X(k-1)) - \hat{u}(k, k-1) \quad (64)$$

where $\hat{u}(k, k-1) = \hat{u}(X(k-1), W_a^{(1)}(k), W_a^{(2)}(k))$ is the output of the actor NN at time $(k-1)$ if the current NN weights are used. It is desired to select the weights of the actor network to minimize the square actor NN error

$$E_a(k) = \frac{1}{2}e_a^2(k). \quad (65)$$

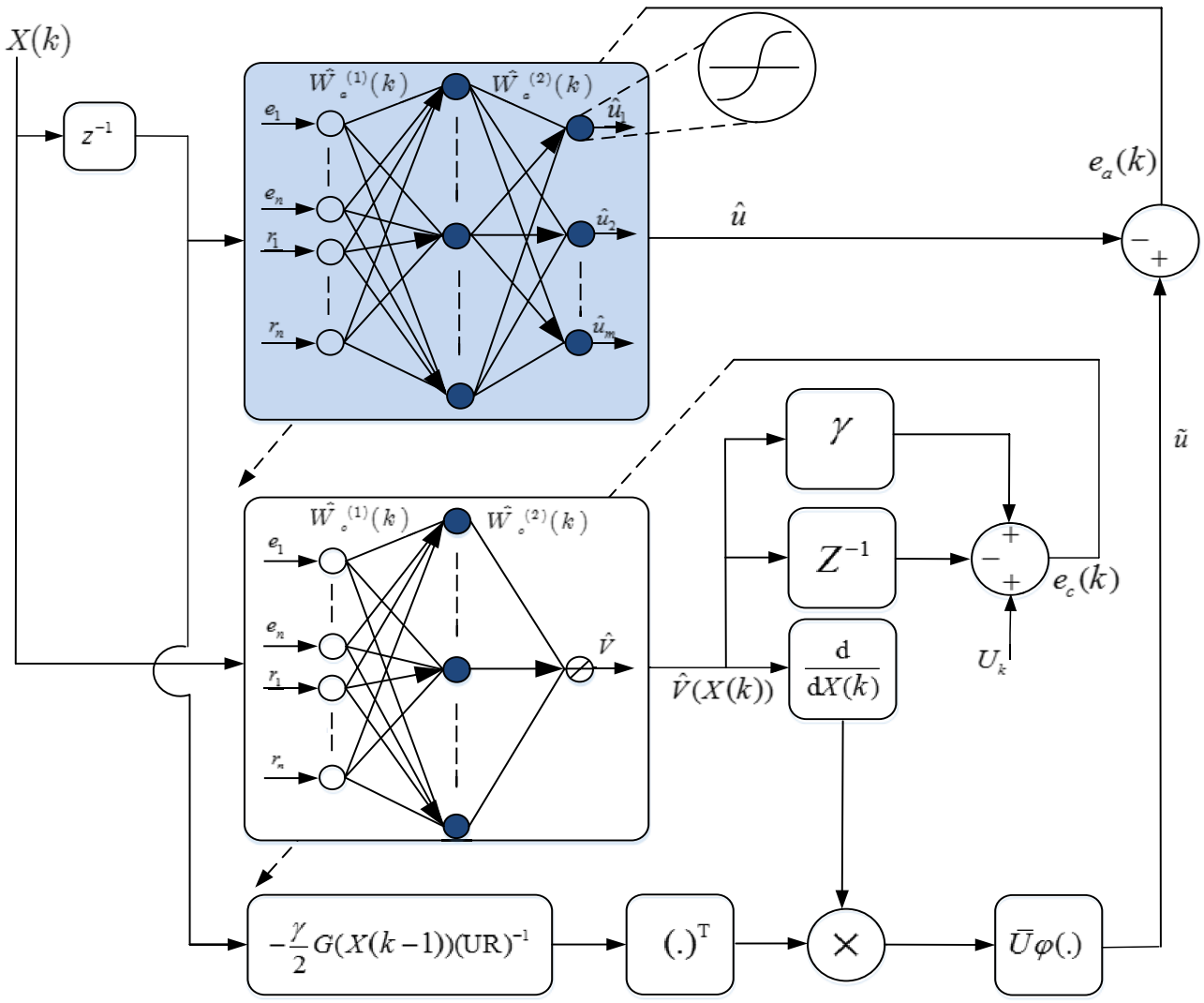

Fig. 3. Schematic of the proposed actor and critic learning.

The tuning laws for the actor weights are selected as the normalized gradient descent algorithm, that is

$$\hat{w}_{alj}^{(1)}(k+1) = \hat{w}_{alij}^{(1)}(k) - l_a \frac{\partial E_a(k)}{\partial \hat{w}_{alj}^{(1)}(k)} \tag{66}$$

$$\hat{w}_{aij}^{(2)}(k+1) = \hat{w}_{aij}^{(2)}(k) - l_a \frac{\partial E_a(k)}{\partial \hat{w}_{aij}^{(2)}(k)} \tag{67}$$

where $l_a > 0$ is learning rate.

The chain rule for the weights between the input layer and the hidden layer yields

$$\begin{aligned}\frac{\partial E_a(k)}{\partial \hat{w}_{aij}^{(1)}(k)} &= \frac{\partial E_a(k)}{\partial e_a(k)} \frac{\partial e_a(k)}{\partial \hat{u}(k,k-1)} \frac{\partial \hat{u}(k,k-1)}{\partial \hat{w}_{aij}^{(1)}(k)} \\ &= e_a(k) - \sum_{l=1}^{m} \Bigg[\bar{u}_l \Big(1 - \frac{1}{\bar{u}_l^2} \sigma_l^2 (X(k-1), W_a^{(1)}(k), \\ &\qquad W_a^{(2)}(k))\Big) \times \sum_{i=1}^{N_a} \hat{w}_{ali}^{(2)}(k) \left(1 - \phi_{al}(k)^2\right) \Bigg] \\ &\quad \times \sum_{j=1}^{2n} X_j(k-1)\end{aligned} \tag{68}$$

and for the weights between the hidden layers and the output layers is

$$\begin{aligned}\frac{\partial E_a(k)}{\partial \hat{w}_{aij}^{(2)}(k)} &= \frac{\partial E_a(k)}{\partial e_a(k)} \frac{\partial e_a(k)}{\partial \hat{u}(k,k-1)} \frac{\partial \hat{u}(k,k-1)}{\partial \hat{w}_{aij}^{(2)}(k)} \\ &= e_a(k) - \bar{u}_i \Big(1 - \frac{1}{\bar{u}_i^2} \sigma_i^2 (X(k-1), W_a^{(1)}(k), \\ &\qquad W_a^{(2)}(k)) \times \phi_j(k)\Big).\end{aligned} \tag{69}$$

Note that the stability proof and convergence of the actor–critic network during learning are provided in [16] and [47].

Fig. 3 shows the schematic of the proposed update law for the actor–critic structure. As can be observed from this figure, the previous values of the state of the system and the state of the system are required to be stored and used in updating the actor and critic NNs weights.

Remark 9: Note that in the proposed algorithm both the actor and critic NNs are updated simultaneously. The control input given by (55) is applied to system continually, while converging to the optimal solution. Since the weight update laws for actor and critic are coupled, the critic NN is also

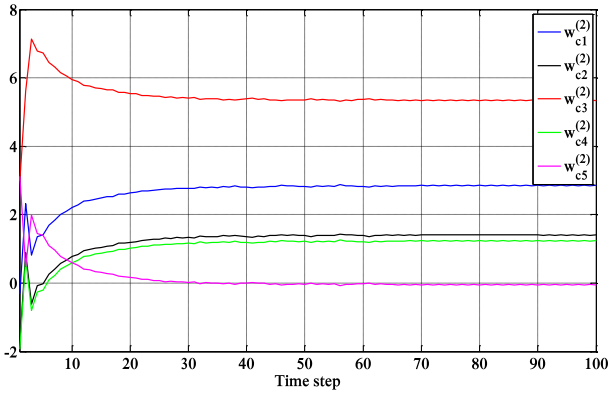

Fig. 4. Weights of the output layer of critic network.

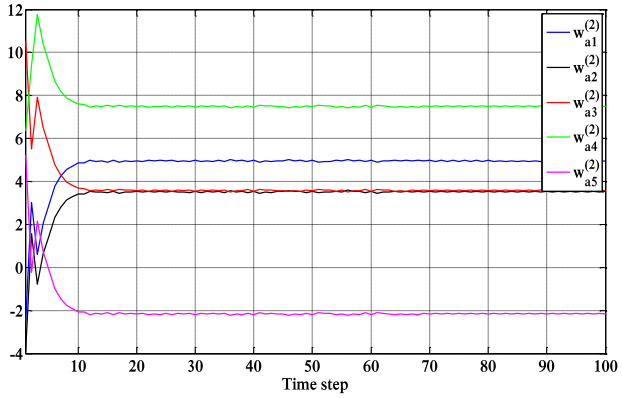

Fig. 5. Weights of the output layer of actor network.

required to be updated continually and simultaneously with the control input. This simultaneous update rule for actor and critic makes convergence guarantees more difficult to prove.

Remark 10: Note that for solving the optimal tracking problem using RL, the control input should be PE. This ensures sufficient exploration of the state of the systems. An exploratory signal consisting of sinusoids of varying frequencies can be added to the control input to ensure PE qualitatively.

V. Simulation Results

In this section, a simulation example is provided to show the design procedures and verify the effectiveness of the proposed scheme.

A nonlinear system dynamics is considered as

$$\begin{aligned} x_1(k+1) &= -0.8\,x_2(k) \\ x_2(k+1) &= -0.45\,x_1(k) - \sin(x_2(k)) + 0.2\,x_2(k)u(k). \end{aligned} \quad (70)$$

It is assumed that the control input is bounded by $|u(k)| \leq 0.4$.

The sinusoid reference trajectory dynamics is generated by the command generator given by

$$\begin{aligned} r_1(k+1) &= -r_1(k) \\ r_2(k+1) &= -r_2(k). \end{aligned} \quad (71)$$

The performance index is consider as (16) with $Q = 20\ I$, $R = 1$, and $\gamma = 0.3$ that I is the identity matrix with appropriate dimension.

The proposed actor–critic algorithm is applied to the system (70). The weights of NN for both actor and critic networks are initialized randomly between -1 and 1. A small probing noise is added to the control input to excite the system during learning.

The critic is a two-layer NN with five activation functions in the hidden layer and one activation function in the output layer. The type of activation functions in the hidden layer is tanh(.) and in the output layer is the identity function.

At the end of learning, the weight matrix between input layer and hidden layer converges to

$$W_c^{(1)} = \begin{bmatrix} -0.9910 & 1.6093 & 5.5495 & 3.8788 \\ 1.1450 & 3.6761 & 2.4605 & 1.9193 \\ 3.0482 & 1.5314 & -1.0640 & -0.2953 \\ 2.1615 & 1.7524 & 2.9390 & 3.2034 \\ -0.8530 & 5.7600 & -1.4361 & -1.8192 \end{bmatrix} \quad (72)$$

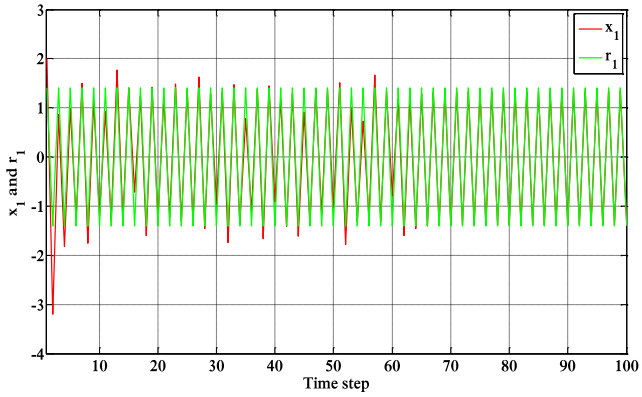

Fig. 6. Evaluation of x_1 and r_1 during the learning process.

and the weight vector between the hidden layer and output layer converges to

$$W_c^{(2)} = \begin{bmatrix} 2.7743 & 1.3367 & 5.2477 & 1.1616 & 0.0228 \end{bmatrix}. \quad (73)$$

Fig. 4 shows that the convergence of the weight vector between the hidden layer and the output layer in the critic network.

The actor is also a two-layer NN with five activation functions in the hidden layer and one $\bar{u}\tanh(.)$ activation function in the output layer that $\bar{u}$ is the bound for control input. The type of the activation functions in the hidden layer is tanh(.).

At the end of learning, the weight matrix between input layer and hidden layer converges to

$$W_a^{(1)} = \begin{bmatrix} 3.0711 & 6.5380 & -1.8147 & 0.9523 \\ -2.9429 & 3.7955 & 5.5219 & 1.2470 \\ -1.4521 & -1.9017 & 2.1629 & 0.8028 \\ 2.2684 & 0.2858 & 3.8524 & 4.7390 \\ 1.6565 & 2.6921 & 3.8270 & 4.9242 \end{bmatrix} \quad (74)$$

and the weight vector between the hidden layer and output layer converges to

$$W_a^{(2)} = \begin{bmatrix} 4.9707 & 3.5331 & 3.5613 & 7.4699 & -2.1780 \end{bmatrix}. \quad (75)$$

Fig. 5 shows that the convergence of the weight vector between the hidden layer and the output layer in the actor network.

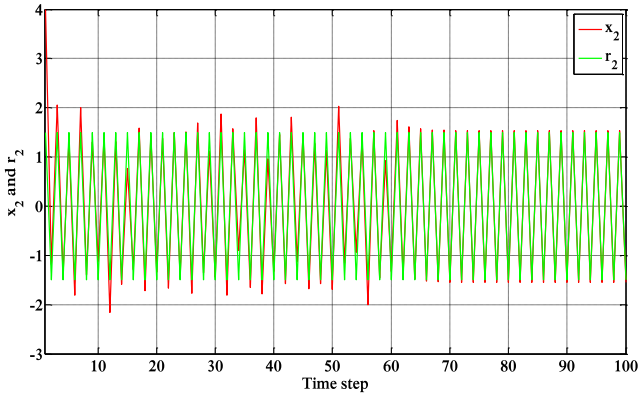

Fig. 7. Evaluation of x_2 and r_2 during the learning process.

Figs. 6 and 7 show that the states of the system $x(k)$ track the reference trajectory $r(k)$ and guarantee the stability for the proposed method after the learning is finished and the probing noise is removed. This confirms that our proposed method successfully finds an optimal tracking controller for system (70).

VI. Conclusion

A new formulation of the nonlinear DT tracking control problem in the presence of input constraints was presented in this paper. A novel discounted performance function was presented and it was shown that the minimization of this performance function gives both feedback and feedforward parts of the bounded optimal control input simultaneously. An actor–critic structure was used to learn the solution of the tracking problem online without requiring knowledge of the system drift dynamics. The actor and critic NNs were updated simultaneously and used previous stored state of the system and value function, instead of their current values, to avoid the requirement of the system model dynamics.

References

[1] W. B. Powell, *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. New York, NY, USA: Wiley, 2009.

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

[3] J. Si, A. G. Barto, W. B. Powell, and D. Wunch, *Handbook of Learning and Approximate Dynamic Programming*. New York, NY, USA: Wiley, 2004.

[4] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 1998.

[5] F. L. Lewis, D. Vrabie, and V. Syrmos, *Optimal Control*, 3rd ed. New York, NY, USA: Wiley, 2012.

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

[7] F. L. Lewis, D. Vrabie, and K. G. Vamvoudakis, "Reinforcement learning and feedback control: Using natural decision methods to design optimal adaptive controllers," *IEEE Control Syst.*, vol. 32, no. 6, pp. 76–105, Dec. 2012.

[8] F. L. Lewis and D. Liu, *Reinforcement Learning and Approximate Dynamic Programming for Feedback Control*. Hoboken, NJ, USA: Wiley, 2013.

[9] M. Fairbank, S. Li, X. Fu, E. Alonso, and D. Wunsch, "An adaptive recurrent neural-network controller using a stabilization matrix and predictive inputs to solve a tracking problem under disturbances," *Neural Netw.*, vol. 49, pp. 74–86, Jan. 2014.

[10] D. Vrabie and F. Lewis, "Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems," *Neural Netw.*, vol. 22, no. 3, pp. 237–246, Apr. 2009.

[11] D. Liu, X. Yang, and H. Li, "Adaptive optimal control for a class of continuous-time affine nonlinear systems with unknown internal dynamics," *Neural Comput. Appl.*, vol. 23, nos. 7–8, pp. 1843–1850, Dec. 2013.

[12] S. Bhasin, R. Kamalapurkar, M. Johnson, K. Vamvoudakis, F. L. Lewis, and W. Dixon, "A novel actor–critic–identifier architecture for approximate optimal control of uncertain nonlinear systems," *Automatica*, vol. 49, no. 1, pp. 82–92, Jan. 2013.

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

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

[15] Y. Luo and H. Zhang, "Approximate optimal control for a class of nonlinear discrete-time systems with saturating actuators," *Prog. Natural Sci.*, vol. 18, no. 8, pp. 1023–1029, 2008.

[16] T. Dierks and S. Jagannathan, "Online optimal control of nonlinear discrete-time systems using approximate dynamic programming," *J. Control Theory Appl.*, vol. 9, no. 3, pp. 361–369, 2011.

[17] J. Si and Y.-T. Wang, "Online learning control by association and reinforcement," *IEEE Trans. Neural Netw.*, vol. 12, no. 2, pp. 264–276, Mar. 2001.

[18] P. J. Werbos, "A menu of designs for reinforcement learning over time," in *Neural Networks for Control*. Cambridge, MA, USA: MIT Press, 1991.

[19] P. J. Werbos, "Approximate dynamic programming for real-time control and neural modeling," in *Handbook of Intelligent Control*, D. A. White and D. A. Sofge, Eds. New York, NY, USA: Reinhold, 1992.

[20] P. J. Werbos, "Neural networks for control and system identification," in *Proc. 28th IEEE CDC*, Dec. 1989, pp. 260–265.

[21] H. Zhang, D. Liu, Y. Luo, and D. Wang, *Adaptive Dynamic Programming for Control: Algorithms and Stability*. London, U.K.: Springer-Verlag, 2012.

[22] Q. Wei and D. Liu, "Data-driven neuro-optimal temperature control of water–gas shift reaction using stable iterative adaptive dynamic programming," *IEEE Trans. Ind. Electron.*, vol. 61, no. 11, pp. 6399–6408, Nov. 2014.

[23] Q. Wei and D. Liu, "A novel iterative θ-adaptive dynamic programming for discrete-time nonlinear systems," *IEEE Trans. Autom. Sci. Eng.*, to be published.

[24] D. Liu and Q. Wei, "Policy iteration adaptive dynamic programming algorithm for discrete-time nonlinear systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 3, pp. 621–634, Mar. 2014.

[25] D. Liu and Q. Wei, "Finite-approximation-error-based optimal control approach for discrete-time nonlinear systems," *IEEE Trans. Cybern.*, vol. 43, no. 2, pp. 779–789, Apr. 2013.

[26] R. Song, W. Xiao, H. Zhang, and C. Sun, "Adaptive dynamic programming for a class of complex-valued nonlinear systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 9, pp. 1733–1739, Sep. 2014.

[27] D. Liu, H. Li, and D. Wang, "Online synchronous approximate optimal learning algorithm for multi-player non-zero-sum games with unknown dynamics," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 44, no. 8, pp. 1015–1027, Aug. 2014.

[28] D. Liu, D. Wang, and H. Li, "Decentralized stabilization for a class of continuous-time nonlinear interconnected systems using online learning optimal control approach," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 2, pp. 418–428, Feb. 2014.

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

[30] H. Modares, F. L. Lewis, and M.-B. Naghibi-Sistani, "Adaptive optimal control of unknown constrained-input systems using policy iteration and neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 10, pp. 1513–1525, Oct. 2013.

[31] H. Modares, F. L. Lewis, and M.-B. Naghibi-Sistani, "Integral reinforcement learning and experience replay for adaptive optimal control of partially-unknown constrained-input continuous-time systems," *Automatica*, vol. 50, no. 1, pp. 193–202, 2014.

[32] H. Zhang, Q. Wei, and Y. Luo, "A novel infinite-time optimal tracking control scheme for a class of discrete-time nonlinear systems via the greedy HDP iteration algorithm," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 38, no. 4, pp. 937–942, Aug. 2008.

[33] T. Dierks and S. Jagannathan, "Optimal tracking control of affine nonlinear discrete-time systems with unknown internal dynamics," in *Proc. 48th IEEE CDC*, Dec. 2009, pp. 6750–6755.
[34] Y. Huang and D. Liu, "Neural-network-based optimal tracking control scheme for a class of unknown discrete-time nonlinear systems using iterative ADP algorithm," *Neurocomputing*, vol. 125, pp. 46–56, Feb. 2014.
[35] H. Zhang, L. Cui, X. Zhang, and X. Luo, "Data-driven robust approximate optimal tracking control for unknown general nonlinear systems using adaptive dynamic programming method," *IEEE Trans. Neural Netw.*, vol. 22, no. 12, pp. 2226–2236, Dec. 2011.
[36] Q. Wei and D. Liu, "Adaptive dynamic programming for optimal tracking control of unknown nonlinear systems with application to coal gasification," *IEEE Trans. Autom. Sci. Eng.*, to be published.
[37] B. Kiumarsi, F. L. Lewis, H. Modares, A. Karimpour, and M.-B. Naghibi-Sistani, "Reinforcement Q-learning for optimal tracking control of linear discrete-time systems with unknown dynamics," *Automatica*, vol. 50, no. 4, pp. 1167–1175, 2014.
[38] B. Kiumarsi-Khomartash, F. L. Lewis, M.-B. Naghibi-Sistani, and A. Karimpour, "Optimal tracking control for linear discrete-time systems using reinforcement learning," in *Proc. IEEE 52nd Annu. CDC*, Florence, Italy, Dec. 2013, pp. 3845–3850.
[39] A. Isidori, *Nonlinear Control Systems*, 3rd ed. London, U.K.: Springer-Verlag, 1995.
[40] S. E. Lyshevski, "Optimal control of nonlinear continuous-time systems: Design of bounded controllers via generalized nonquadratic functionals," in *Proc. IEEE ACC*, Jun. 1998, pp. 205–209.
[41] K. Doya, "Reinforcement learning in continuous time and space," *Neural Comput.*, vol. 12, no. 1, pp. 219–245, 2000.
[42] M. Fairbank, "Value-gradient learning," Ph.D. dissertation, Dept. Comput. Sci., City Univ. London, London, U.K., 2014.
[43] F. L. Lewis, S. Jagannathan, and A. Yesildirek, *Neural Network Control of Robot Manipulators and Nonlinear Systems*. New York, NY, USA: Taylor & Francis, 1999.
[44] K. G. Vamvodakis and F. L. Lewis, "Online actor–critic algorithm to solve the continuous-time infinite horizon optimal control problem," *Automatica*, vol. 46, no. 5, pp. 878–888, 2010.
[45] P. J. Werbos. (1998). "Stable adaptive control using new critic designs." [Online]. Available: http://arxiv.org/abs/adap-org/9810001
[46] L. Baird, "Residual algorithms: Reinforcement learning with function approximation," in *Proc. 12th Int. Conf. Mach. Learn.*, 1995, pp. 30–37.
[47] X. Luo and J. Si, "Stability of direct heuristic dynamic programming for nonlinear tracking control using PID neural network," in *Proc. IJCNN*, Dallas, TX, USA, Aug. 2013, pp. 1–7.

Bahare Kiumarsi (S'14) received the B.S. degree from the Shahrood University of Technology, Shahrud, Iran, in 2009, and the M.S. degree from the Ferdowsi University of Mashhad, Mashhad, Iran, in 2013. She is currently pursuing the Ph.D. degree with the University of Texas at Arlington (UTA), Fort Worth, TX, USA.

She was a Visiting Scholar with UTA from 2012 to 2013. Her current research interests include optimal control, reinforcement learning, and neural networks.

Frank L. Lewis (S'70–M'81–SM'86–F'94) received the bachelor's degree in physics/electrical engineering and the M.S.E.E. degree from Rice University, Houston, TX, USA, the M.S. degree in aeronautical engineering from the University of West Florida, Pensacola, FL, USA, and the Ph.D. degree from the Georgia Institute of Technology, Atlanta, GA, USA.

He is currently a Distinguished Scholar Professor and Distinguished Teaching Professor with the University of Texas at Arlington, Fort Worth, TX, USA, the Moncrief-O'Donnell Chair of the University of Texas at Arlington Research Institute, Fort Worth, the Qian Ren Thousand Talents Professor and Project 111 Professor with Northeastern University, Shenyang, China, and a Distinguished Visiting Professor with the Nanjing University of Science and Technology, Nanjing, China. He is involved in feedback control, intelligent systems, cooperative control systems, and nonlinear systems. He has authored six U.S. patents, numerous journal special issues, journal papers, and 20 books, including *Optimal Control*, *Aircraft Control*, *Optimal Estimation*, and *Robot Manipulator Control*, which are used as university textbooks worldwide.

Prof. Lewis is a member of the National Academy of Inventors, a fellow of the International Federation of Automatic Control and the Institute of Measurement and Control, U.K., a Texas Board of Professional Engineer, a U.K. Chartered Engineer, and a founding member of the Board of Governors of the Mediterranean Control Association. He was a recipient of the Fulbright Research Award, the NSF Research Initiation Grant, the Terman Award from the American Society for Engineering Education, the Gabor Award from the International Neural Network Society, the Honeywell Field Engineering Medal from the Institute of Measurement and Control, U.K., the Neural Networks Pioneer Award from IEEE Computational Intelligence Society, the Outstanding Service Award from the Dallas IEEE Section, and the Texas Regents Outstanding Teaching Award in 2013. He was elected as an Engineer of the year by the Fort Worth IEEE Section. He was listed in the Fort Worth Business Press Top 200 Leaders in Manufacturing.