Reinforcement Learning Controller Design for Affine Nonlinear Discrete-Time Systems using Online Approximators

Qinmin Yang and Sarangapani Jagannathan

***Abstract*—In this paper, reinforcement learning state- and output-feedback-based adaptive critic controller designs are proposed by using the online approximators (OLAs) for a general multi-input and multioutput affine unknown nonlinear discrete-time systems in the presence of bounded disturbances. The proposed controller design has two entities, an action network that is designed to produce optimal signal and a critic network that evaluates the performance of the action network. The critic estimates the cost-to-go function which is tuned online using recursive equations derived from heuristic dynamic programming. Here, neural networks (NNs) are used both for the action and critic whereas any OLAs, such as radial basis functions, splines, fuzzy logic, etc., can be utilized. For the output-feedback counterpart, an additional NN is designated as the observer to estimate the unavailable system states, and thus, separation principle is not required. The NN weight tuning laws for the controller schemes are also derived while ensuring uniform ultimate boundedness of the closed-loop system using Lyapunov theory. Finally, the effectiveness of the two controllers is tested in simulation on a pendulum balancing system and a two-link robotic arm system.**

***Index Terms*—Adaptive critic, dynamic programming (DP), Lyapunov method, neural networks (NNs), online approximators (OLAs), online learning, reinforcement learning.**

I. Introduction

IN THE literature, there are many ways for designing stable controllers for nonlinear dynamic systems. However, stability is only a bare requirement for the controller design. A further consideration is the optimality based on a predefined cost function, which is used to determine the performance of the system. In other words, a controller scheme should not only achieve the stability of the closed-loop system but also keep the cost as small as possible. A number of methods have been introduced for the optimal control of nonlinear systems.

Of the available methods, dynamic programming (DP) has been extensively applied to generate optimal control for nonlinear dynamic systems [1]–[4], [22] by utilizing Bellman's principle of optimality that is stated as "no matter how an intermediate point is reached in an optimal trajectory, the rest of the trajectory (from the intermediate point to the end) must be optimal." While this method allows optimization over time [17], one of its drawbacks is the computation cost with the increasing dimension of the nonlinear system, which is referred to as the "curse of dimensionality" [4]. Therefore, to confront this issue, adaptive or approximate DP (ADP) methods (e.g., see [8] and [29]) have been developed recently. However, most of them are implemented either by an offline manner using iterative schemes or require the dynamics of the nonlinear system to be known *a priori*. Unfortunately, these requirements are often not practical for real-world systems, since the exact dynamics is usually not available for nonlinear systems.

On the other hand, reinforcement learning is originated from animal behavior research and its interactions with the environment. It is based on the common sense reasoning that, if an action is followed by a satisfactory outcome (reinforcement signal), then the tendency to repeat that action is strengthened, i.e., reinforced. Differing from the traditional supervised neural network (NN) learning, there is no desired behavior or training examples employed with reinforcement learning schemes. Nevertheless, it is common to apply reinforcement learning for optimal controller design, since the cost function can be directly seen as a form of reinforcement signal.

Of the available reinforcement learning schemes, the temporal difference learning method [9]–[12] and ADP [31] have found many applications in engineering since it does not require the knowledge of the system dynamics even though an iterative approach is typically utilized. However, to obtain a satisfactory reinforcement signal for each action, the approach must visit each system state by applying each action often enough [13], which requires that the system be time invariant, or stationary in the case of stochastic system.

To overcome the iterative offline methodology for real-time applications, several appealing online approximator (OLA)-based controller design methods were introduced in [15], [17], [19]–[22], and [24]. They are also referred to as forward DP or adaptive critic designs (ACDs). The central theme of this approach is that the optimal control law and cost function are approximated by online parametric structures, such as NNs, which are trained over time along with the information that is fed back from the system response. Depending upon whether the NNs approximate the cost function or its derivative, or both, the ACDs are classified as follows: 1) heuristic DP (HDP); 2) dual heuristic programming (DHP);

Manuscript received October 18, 2010; revised May 8, 2011; accepted August 13, 2011. Date of publication September 22, 2011; date of current version March 16, 2012. This work was supported in part by NSF under Grants ECCS 0624644 and ECCS 0901562 by NSF of China, under Grant 60804064, and by the Qianjiang Program of Zhejiang Province under Grant 2011R10024. This paper was recommended by Associate Editor W. J. Wang.

Q. Yang is with the State Key Laboratory of Industrial Control Technology, Department of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China (e-mail: qmyang@iipc.zju.edu.cn).

S. Jagannathan is with the Department of Electrical and Computer Engineering, Missouri University of Science and Technology (formerly, University of Missouri–Rolla), Rolla, MO 65409, USA.

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/TSMCB.2011.2166384

and 3) globalized dual heuristic programming (GDHP). It is important to note that, when the action is introduced as an additional input to the critic, then the ACD will be referred as action dependent version of the ACD.

Instead of finding the exact minimum, ACDs approximate the Bellman equation $J(x(k)) = \min_{u(k)}\{J(x(k+1)) + U(x(k), x(k+1))\}$ where $x(k)$ is the state and $u(k)$ is the control at time step k, while the strategic utility function $J(x(k))$ represents the minimum cost or performance measure associated with going from k to final step N, $U(x(k), x(k+1))$ is the utility function denoting the cost incurred in going from k to $k+1$ using control $u(k)$, and $J(k+1)$ is the minimum cost or performance measure associated in going from state $k+1$ to the final step N. The NNs are widely used for approximation in the ACD literature.

Numerous papers have appeared on ACDs using NN, but stability is rarely studied. The ones where some sort of stability is discussed are briefly introduced next. A new NN learning algorithm based on gradient descent rule is introduced in [6], and the approach is evaluated on a single cart-pole balancing system and a pendulum and a triple-link inverted pendulum. However, no proof of the convergence or stability of the system was given. By contrast, Lyapunov analysis was derived in [14] and [15] using a variant of Bellman equation.

On the other hand, [6], [14], and [15] employ simplified binary reward or cost function which is a variant of the standard Bellman equation. By contrast, in this paper, a novel -reinforcement-learning-based controller is introduced for multi-input multioutput affine nonlinear discrete-time systems with standard Bellman equation first by assuming state feedback using NN. However, the approach is generic enough that any OLA-based scheme, such as radial basis functions, splines, and Cerebellar Model Articulation Controller (CMAC), can be utilized.

Meanwhile, most of the developed methods in ADP and reinforcement learning schemes [24], [31] use state feedback. However, it is common in practical applications that outputs are measured and the system states are unavailable for measurement. An observer is used for estimating the states by using the measured outputs. For nonlinear systems, a stable state observer, in general, does not guarantee the stability of the entire closed-loop system when it is used in conjunction with a stabilizing controller. While the separation principle holds for linear systems, it does not hold for nonlinear systems [28], and the design of an observer-based controller design becomes more challenging if optimality has to be ensured. Therefore, an output-feedback controller with reinforcement learning design is also proposed in this paper.

In this paper, ONA-based methodology by using NNs is considered for the control of nonlinear discrete systems with standard quadratic-performance index as the cost function. The state-feedback scheme consists of two OLAs (in this case, two NNs): an action NN for the action network to derive the optimal (or near optimal) control signal to track not only the desired system output but also to minimize the long-term cost function and a critic NN for the critic network to approximate the long-term cost function $J(x(k))$ and to tune the action NN weights. For the output-feedback controller, an additional NN is employed as the observer to estimate the unavailable system states.

Since the control signal is not used in the critic NN as an additional input, the proposed approach could be seen as an OLA-based HDP or neural HDP approach. Other than addressing the problem of optimization, contributions of this paper include the following: 1) the demonstration of the uniformly ultimately boundedness (UUB) of the overall system even in the presence of approximation errors and bounded unknown disturbances unlike in the existing adaptive critic works where the convergence is shown under ideal circumstances [27]; 2) the NN weights are tuned online instead of offline training that is commonly employed in ACD [31]; and 3) the linear-in-parameter assumption is overcome along with the persistent excitation (PE) condition requirement [26]. Finally, the proposed approach uses the standard Bellman equation and not a variant of the Bellman equation [14], [15].

This paper is organized as follows. In Section II, the background of ACDs and assumptions of the system are introduced. Section III presents the detailed state-feedback controller design methodology with learning algorithm for the action and critic NNs. Output-feedback control scheme is proposed in Section IV, and simulation results on two-link robotic arm and pendulum are demonstrated in Section V.

II. Background

A. ACD

In this paper, consider the following nonlinear affine system, given in the form as

$$\begin{aligned} x_1(k+1) &= x_2(k) \\ &\vdots \\ x_n(k+1) &= f\left(x(k)\right) + g\left(x(k)\right)u(k) + d(k) \\ y(k) &= x_1(k) \end{aligned} \tag{1}$$

where the state vector $x(k) = [x_1(k),\ x_2(k),\ \ldots,\ x_n(k)]^T \in \Re^{nm}$ at time instant k with each $x_i(k) \in \Re^m$, $i = 1, \ldots, n$, $f(x(k)) \in \Re^m$ is a smooth unknown nonlinear vector field, $g(x(k)) \in \Re^{m\times m}$ is a diagonal matrix of unknown nonlinear vector fields, $u(k) \in \Re^m$ is the control input vector, $y(k) \in \Re^m$ is the output vector, and $d(k) \in \Re^m$ is the unknown but bounded disturbance vector, whose bound is assumed to be a known constant such that $\|d(k)\| \le d_m$. Here, $\| \bullet \|$ stands for the Frobenius norm [5], which will be used throughout this paper. First, the state vector $x(k)$ is assumed available at the kth step for the state-feedback controller.

Assumption 1: Without loss of generality, let the matrix $g(x(k)) \in \Re^{m\times m}$ be a positive definite diagonal matrix for each $x(k) \in \Re^{nm}$, with $g_{\min} \in \Re^+$ and $g_{\max} \in \Re^+$ represent the minimum and maximum eigenvalues of the matrix $g(x(k))$, respectively, such that $0 < g_{\min} \le g_{\max}$.

Our objective is to design an online reinforcement learning NN controller for the system (1) such that the following holds: 1) all the signals in the closed-loop system remain uniformly ultimately bounded in the presence of bounded disturbances and approximation errors; 2) the state $x(k)$ follows a desired trajectory $x_d(k) = [x_{1d}(k),\ \ldots,\ x_{nd}(k)]^T \in \Re^{nm}$, or, equivalently speaking, the output $y(k)$ follows a desired trajectory $y_d(k) \in \Re^m$; and 3) the long-term cost function (2) is minimized so that an optimal (or near optimal) control input can be generated. Here, the "online" means that the controller learns "in real time" through constant interaction with the instant, instead of an explicit offline learning phase.

Assumption 2: The desired trajectory of the system states $x_d(k) = [x_{1d}(k),\ \ldots,\ x_{nd}(k)]^T$ is arbitrarily selected, but satisfies $x_{id}(k+1) = x_{(i+1)d}(k)$, $i = 1, \ldots, n-1$, and $y_d(k)$ is a known smooth bounded function over the compact subset of $\Re^m$.

Note that, from Assumption 2 and (1), one can derive that $x_{id}(k) = y_d(k+i-1)$, $i = 1, \ldots, n$. Meanwhile, to introduce optimality into our design, the long-term cost function is defined as

$$\begin{aligned} J(k) = J\left(x(k), u\right) &= \sum_{i=t_0}^{\infty} \gamma^i r(k+i) \\ &= \sum_{i=t_0}^{\infty} \gamma^i \left[q\left(x(k+i)\right) + u^T(k+i)Ru(k+i)\right] \end{aligned} \quad (2)$$

where $J(k)$ stands for $J(x(k), u)$ for simplicity, u is a control policy, R is a positive definite matrix, and $q(x(k))$ is a positive semidefinite function of the states, while $\gamma (0 \le \gamma \le 1)$ is the discount factor for the infinite-horizon problem. As observed from (2), the long-term cost is defined in terms of the discounted sum of the immediate cost or Lagrangian $r(k)$, which is given by

$$\begin{aligned} r(k) &= q\left(x(k)\right) + u^T(k)Ru(k) \\ &= \left(x_1(k) - x_{1d}(k)\right)^T Q \left(x_1(k) - x_{1d}(k)\right) + u^T(k)Ru(k) \\ &= \left(y(k) - y_d(k)\right)^T Q \left(y(k) - y_d(k)\right) + u^T(k)Ru(k) \end{aligned} \quad (3)$$

where Q is a positive semidefinite matrix. In this paper, we are using a widely used standard quadratic cost function defined based on the output tracking error $e(k) = y(k) - y_d(k)$, which will be contrasted with [6], [14], and [15]. The immediate cost function $r(k)$ can be viewed as a system performance index for the current step.

The basic idea in adaptive critic or reinforcement learning design is to approximate the long-term cost function J (or its derivative, or both) and generate the control signal minimizing the cost. By using learning, the OLA will converge to the optimal cost, and the controller will converge to the optimal controller correspondingly. In fact, for an optimal control law, which can be expressed as $u^*(k) = u^*(x(k))$, the optimal long-term cost function can be written alternatively as $J^*(k) = J^*(x(k), u^*(x(k))) = J^*(x(k))$, which is a function of the current state [16]. Next, one can state the following reasonable assumption.

Assumption 3: The optimal cost function $J^*(k)$ is finite and bounded over the compact set $S \subset \Re^n$ by J_m.

B. Two-Layer NN

In our controller architecture, we consider the NNs having two layers, as shown in Fig. 1. The output of the NN can be given by $Y = W^T\phi(V^T X)$, where V and W are the hidden layer and output layer weights, respectively. The number of hidden layer nodes is denoted as N_2. A general function $f(x) \in C^{N_3}(S)$ can be written as [18]

$$f(x) = W^T\phi(V^T x) + \varepsilon(x) \quad (4)$$

with $\varepsilon(x)$ being an NN functional reconstruction error vector. In our design, V is selected initially at random and held. It is demonstrated in [18] that, if the hidden layer weights V are chosen initially at random and held while N_2 is sufficiently large, the NN approximation error $\varepsilon(x)$ can be made arbitrarily small since the activation function vector forms a basis. Additionally, one can relax this assumption of bounded approximation error by using a robust signal through an auxiliary control input, which is relegated as part of a future publication. Since NN is utilized here as an illustration of the OLA, the rest of this paper uses NNs for the OLA.

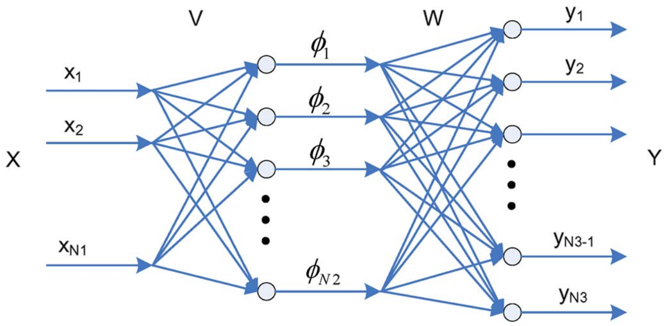


Fig. 1. Two-layer NN structure.

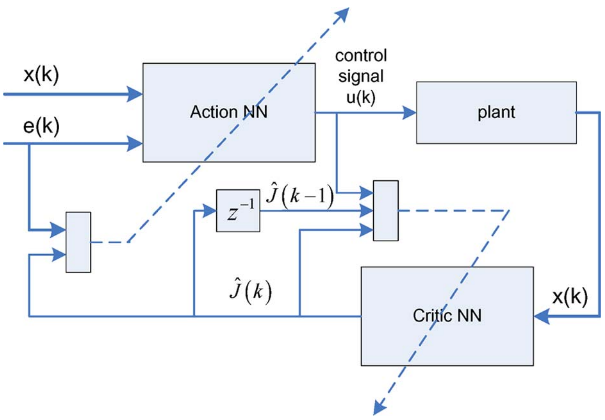


Fig. 2. Online neural DP-based controller structure.

III. State-Feedback Online Reinforcement Learning Controller Design

The block diagram of the proposed controller is shown in Fig. 2 where the action NN is providing the control signal to the plant while the critic NN approximates the long-term cost function. The two NNs learn online without any offline learning phase.

Furthermore, the persistence of excitation (PE) condition is usually necessary to guarantee boundedness of the NN weight estimates [5], [25]. Unfortunately, it is difficult to verify the PE condition of the output function $\phi(V^T x)$ of the NN hidden layer in practical applications. In this paper, a novel tuning algorithm is therefore proposed to make the NN weights robust so that PE is not needed. First, the action NN design is given, and subsequently, the critic NN design is introduced for the state-feedback controller. Next, the weight tuning updates are presented before analyzing the stability of the overall system.

A. Action Network Design

The tracking error at instant k is defined as

$$\begin{aligned} e_i(k) = x_i(k) - x_{id}(k) = x_i(k) - y_d(k+i-1),\\ i = 1, \ldots, n. \end{aligned} \tag{5}$$

Then, future value of the tracking error using system dynamics from (1) can be rewritten as

$$\begin{aligned} e_n(k+1) =& f\left(x(k)\right) + g\left(x(k)\right) u(k) +\\ & d(k) - y_d(k+n). \end{aligned} \tag{6}$$

A desired control signal can be given by

$$u_d(k) = g^{-1}\left(x(k)\right)\left(-f\left(x(k) + y_d(k+n) + l_1 e(k)\right)\right. \tag{7}$$

where $l_1 \in \Re^{m\times m}$ is a design matrix selected such that the tracking error $e_n(k)$ converges to zero.

Since both of $f(x(k))$ and $g(x(k))$ are unknown smooth nonlinear functions, the desired feedback control $u_d(k)$ cannot be implemented directly. Instead, an action NN is employed to generate the control signal. From (7) and considering Assumption 2, the desired control signal can be approximated as

$$\begin{aligned} u_d(k) &= w_a^T \phi_a\left(v_a^T s(k)\right) + \varepsilon_a\left(s(k)\right)\\ &= w_a^T \phi_a\left(s(k)\right) + \varepsilon_a\left(s(k)\right) \end{aligned} \tag{8}$$

where $s(k) = [x^T(k), y_d^T(k), y_d^T(k+n)]^T \in \Re^{(n+2)m}$ is the action NN input vector. The action NN consists of two layers, and $w_a \in \Re^{n_a\times m}$ and $v_a \in \Re^{(n+2)m\times n_a}$ denote the desired weights of the output and hidden layers, respectively, with $\varepsilon_a(s(k))$ being the action NN approximation error and n_a being the number of the neurons in the hidden layer. Since v_a is fixed, for simplicity purposes, the hidden layer activation function vector $\phi_a(v_a^T s(k)) \in \Re^{n_2}$ is denoted as $\phi_a(s(k))$.

Considering the fact that the target weights of the action NN are unknown, the actual NN weights have to be trained online, and its actual output can be expressed as

$$u(k) = \hat{w}_a^T(k)\phi_a\left(v_a^T s(k)\right) = \hat{w}_a^T(k)\phi_a\left(s(k)\right) \tag{9}$$

where $\hat{w}_a(k) \in \Re^{n_a\times m}$ is the actual weight matrix of the output layer at instant k.

Using the action NN output as the control signal and substituting (8) and (9) into (6) yield

$$\begin{aligned} e_n(k+1) &= f\left(x(k)\right) + g\left(x(k)\right) u(k) + d(k) - y_d(k+n)\\ &= l_1 e_n(k) + g\left(x(k)\right)\left(u(k) - u_d(k)\right) + d(k)\\ &= l_1 e_n(k) + g\left(x(k)\right)\\ &\quad \times \left(\tilde{w}_a^T(k)\phi_a\left(s(k)\right) - \varepsilon_a\left(s(k)\right)\right) + d(k)\\ &= l_1 e_n(k) + g\left(x(k)\right)\zeta_a(k) + d_a(k) \end{aligned} \tag{10}$$

where

$$\tilde{w}_a(k) = \hat{w}_a(k) - w_a \tag{11}$$

$$\zeta_a(k) = \tilde{w}_a^T(k)\phi_a\left(s(k)\right) \tag{12}$$

$$d_a(k) = -\, g\left(x(k)\right)\varepsilon_a\left(s(k)\right) + d(k). \tag{13}$$

Thus, the closed-loop tracking error dynamics is expressed as

$$e_n(k+1) = l_1 e_n(k) + g\left(x(k)\right)\zeta_a(k) + d_a(k). \tag{14}$$

In the meantime, we can notice that $e_i(k) = x_i(k) - x_{id}(k) = e_n(k-n+i)$, $i = 1, \ldots, n$. Next, the critic NN design is introduced.

B. Critic Network Design

As stated earlier, an optimal controller should be able to stabilize the closed-loop system while minimizing the cost function at the same time. In this paper, a critic NN is employed to approximate the target long-term cost function $J(k)$. Since $J(k)$ is unavailable at the kth time instant in an online learning framework, the critic NN is tuned online in order to ensure that its output converges close to $J(k)$.

First, the prediction error for the critic or the Bellman error [6] is defined as

$$e_c(k) = \gamma\hat{J}(k) - \hat{J}(k-1) + r(k) \tag{15}$$

where the subscript "c" stands for the "critic" and

$$\hat{J}(k) = \hat{w}_c^T(k)\phi_c\left(v_c^T x(k)\right) = \hat{w}_c^T(k)\phi_c\left(x(k)\right) \tag{16}$$

with $\hat{J}(k) \in \Re$ representing the critic NN output which is an approximation of $J(k)$. In our design, the critic NN is also a two-layer NN, while $\hat{w}_c(k) \in \Re^{n_c\times 1}$ and $v_c \in \Re^{nm\times n_c}$ represent its actual weight matrix of the output and hidden layers, respectively. The term n_c denotes the number of the neurons in the hidden layer. Similar to HDP, the system states $x(k) \in \Re^n$ are selected as the critic network input. The activation function vector of the hidden layer $\phi_c(v_c^T x(k)) \in \Re^{n_c}$ is denoted as $\phi_c(x(k))$ for simplicity. Provided that enough number of the neurons are in the hidden layer, the optimal long-term cost function $J^*(k)$ can be approximated by the critic network with arbitrarily small approximation error $\varepsilon_c(k)$ as

$$\begin{aligned} J^*(k) =& w_c^T\phi_c\left(v_c^T x(k)\right) + \varepsilon_c\left(x(k)\right)\\ =& w_c^T\phi_c\left(x(k)\right) + \varepsilon_c\left(x(k)\right). \end{aligned} \tag{17}$$

Similarly, the critic network NN weight estimation error can be defined as

$$\tilde{w}_c(k) = \hat{w}_c(k) - w_c \tag{18}$$

where the approximation error is given by

$$\zeta_c(k) = \tilde{w}_c^T(k)\phi_c\left(x(k)\right). \tag{19}$$

Thus, we have

$$\begin{aligned} e_c(k) &= \gamma\hat{J}(k) - \hat{J}(k-1) + r(k)\\ &= \gamma\zeta_c(k) + \gamma J^*(k) - \zeta_c(k-1)\\ &\quad - J^*(k-1) + r(k) - \varepsilon_c(k) + \varepsilon_c(k-1). \end{aligned}$$

Next, we discuss the weight tuning algorithms for critic and action NNs.

C. Weight Update for the Critic Network

Following the discussion from the last section, the objective function to be minimized by the critic network can be defined as a quadratic function of tracking errors as

$$E_c(k) = \frac{1}{2}e_c^T(k)e_c(k) = \frac{1}{2}e_c^2(k). \tag{20}$$

Using a standard gradient-based adaptation method, the weight updating algorithm for the critic network is given by

$$\hat{w}_c(k+1) = \hat{w}_c(k) + \Delta\hat{w}_c(k) \tag{21}$$

where

$$\Delta\hat{w}_c(k) = \alpha_c\left[-\frac{\partial E_c(k)}{\partial\hat{w}_c(k)}\right] \tag{22}$$

with $\alpha_c \in \Re$ being the adaptation gain.

Combining (15), (16), and (20) with (22), the critic NN weight updating rule can be obtained by using the chain rule as

$$\begin{aligned}\Delta\hat{w}_c(k) =& -\alpha_c\frac{\partial E_c(k)}{\partial\hat{w}_c(k)} = -\alpha_c\frac{\partial E_c(k)}{\partial e_c(k)}\frac{\partial e_c(k)}{\partial\hat{J}(k)}\frac{\partial\hat{J}(k)}{\partial\hat{w}_c(k)}\\ =& -\alpha_c\gamma\phi_c\left(x(k)\right)e_c(k).\end{aligned} \tag{23}$$

Thus, the critic NN weight updating algorithm is obtained as

$$\begin{aligned}\hat{w}_c(k+1) = \hat{w}_c(k) - \alpha_c\gamma\phi_c\left(x(k)\right)&\\ \left(\gamma\hat{J}(k) + r(k) - \hat{J}(k-1)\right).&\end{aligned} \tag{24}$$

D. Weight Update for the Action Network

The basis for adapting the action NN is to track the desired trajectory and to lower the cost function. Therefore, the error for the action NN can be formed by using the functional estimation error $\zeta_a(k)$ and the error between the nominal desired long-term cost function $J_d(k) \in \Re$ and the critic signal $\hat{J}(k)$. Now, we define the cost function vector as $\bar{J}(k) = [\hat{J}(k) \quad \hat{J}(k) \quad \ldots \quad \hat{J}(k)]^T \in \Re^{m\times 1}$. Let

$$\begin{aligned}e_a(k) &= \sqrt{g\left(x(k)\right)}\zeta_a(k) + \left(\sqrt{g\left(x(k)\right)}\right)^{-1}\left(\bar{J}(k) - J_d(k)\right)\\ &= \sqrt{g\left(x(k)\right)}\zeta_a(k) + \left(\sqrt{g\left(x(k)\right)}\right)^{-1}\bar{J}(k)\end{aligned} \tag{25}$$

where $\zeta_a(k)$ is defined in (12). Given Assumption 1, we define $\sqrt{g(x(k))} \in \Re^{m\times m}$ as the principle square root of the diagonal positive definite matrix $g(x(k))$, i.e., $\sqrt{g(x(k))} \times \sqrt{g(x(k))} = g(x(k))$, and $(\sqrt{g(x(k))})^T = \sqrt{g(x(k))}$ [14]. The desired long-term cost function $J_d(k)$ is nominally defined and is considered to be zero (“0”), which means as low as possible.

Hence, the weights of the action NN $\hat{w}_a(k)$ are tuned to minimize the error

$$E_a(k) = \frac{1}{2}e_a^T(k)e_a(k). \tag{26}$$

Combining (10), (12), (14), (25), and (26) and using the chain rule yield

$$\begin{aligned}\Delta\hat{w}_a(k) =& -\alpha_a\frac{\partial E_a(k)}{\partial\hat{w}_a(k)} = -\alpha_a\frac{\partial E_a(k)}{\partial e_a(k)}\frac{\partial e_a(k)}{\partial\zeta_a(k)}\frac{\partial\zeta_a(k)}{\partial\hat{w}_c(k)}\\ =& -\alpha_a\phi_a\left(s(k)\right)\left(g\left(x(k)\right)\zeta_a(k) + \bar{J}(k)\right)^T\\ =& -\alpha_a\phi_a\left(s(k)\right)\\ &\times\left(e_n(k+1) - l_1e_n(k) - d_a(k) + \bar{J}(k)\right)^T\end{aligned} \tag{27}$$

where $\alpha_a \in \Re^+$ is the adaptation gain of the action NN. However, $d_a(k)$ is typically unavailable, so as in the ideal case, we take it as zero and obtain the weight updating algorithm for the action NN as

$$\begin{aligned}\hat{w}_a(k+1) =& \hat{w}_a(k) - \alpha_a\phi_a\left(s(k)\right)\left(e_n(k+1)\right.\\ &\left. - l_1e_n(k) + \bar{J}(k)\right)^T.\end{aligned} \tag{28}$$

E. Theoretic Result

Assumption 4: Let w_a and w_c be the unknown output layer target weights for the action and critic NNs, respectively, and assume that they are upper bounded such that

$$\|w_a\| \le w_{am} \text{ and } \|w_c\| \le w_{cm} \tag{29}$$

where $w_{am} \in \Re^+$ and $w_{cm} \in \Re^+$ represent the bounds on the unknown target weights.

Fact 1: The activation functions for the action and critic NNs are bounded by known positive values, such that

$$\|\phi_a(k)\| \le \phi_{am} \qquad \|\phi_c(k)\| \le \phi_{cm} \tag{30}$$

where ϕ_{am}, $\phi_{cm} \in \Re^+$ is the upper bound for the activation functions.

Assumption 5: The NN approximation errors $\varepsilon_a(s(k))$ and $\varepsilon_c(x(k))$ are bounded above over the compact set $S \subset \Re^m$ by ε_{am} and ε_{cm} [11].

Fact 2: With the Assumptions 1 and 4, the term $d_a(k)$ in (13) is bounded over the compact set $S \subset \Re^m$ by

$$\|d_a(k)\| \le d_{am} = g_{\max}\varepsilon_{am} + d_m. \tag{31}$$

Combining Assumptions 1, 3, and 4 and Facts 1 and 2, the following theorem is introduced.

Theorem 1: Consider the nonlinear affine system given by (1) with all system states measurable. Let the Assumptions 1–5 hold with the disturbance bound d_m being a known constant. Let the control input be provided by the action NN (9), with the critic NN input being (16). Furthermore, let the weights of the action NN and the critic NN be tuned by (23) and (27), respectively. Then, there exist three positive constants b_e, b_a, and b_c, denoted as the uniform ultimate bounds for the tracking error $e(k)$, and the NN weight estimates the errors of the action and critic NNs $\zeta_a(k)$ and $\zeta_c(k)$, respectively, given by

$$b_e = 2\sqrt{3}D_M/\sqrt{4\gamma_1\left(1 - 3l_{\max}^2\right) - 3\gamma_3Q_{\max}} \tag{32}$$

$$b_a = 2\sqrt{2}D_M/\sqrt{8\gamma_2g_{\min} - 8\gamma_1g_{\max}^2 - \gamma_3R_{\max}} \tag{33}$$

$$b_c = D_M/\sqrt{\gamma_3\gamma^2 - \gamma_2'n - \gamma_4} \tag{34}$$

provided that the controller design parameters satisfy

$$0 < \alpha_a\|\phi_a(k)\|^2 < \frac{g_{\min}}{g_{\max}^2} \tag{35}$$

$$0 < \alpha_c\gamma^2\|\phi_c(k)\|^2 < 1 \tag{36}$$

$$0 < l_{1\max} < \sqrt{3}/3 \tag{37}$$

$$\gamma > 1/2 \tag{38}$$

where α_a and α_c are NN adaptation gains, γ is employed to define the strategic utility function, and $l_{1\max} \in \Re^+$ is the largest eigenvalue of square matrix l_1.

Proof: See the Appendix.

Remark 1: The action and critic NN weights can be initialized at zero or random. This means that there is no explicit offline learning phase needed.

Remark 2: It is important to note that persistency of excitation condition is not utilized, and certainty equivalence (CE) principle is not employed, in contrast to standard work in discrete-time adaptive control [30]. In the latter, a parameter identifier is first designed, and the parameter estimation errors are shown to converge to small values by using a Lyapunov function. Then, in the tracking proof, it is assumed that the parameter estimates are exact by invoking a CE assumption, and another Lyapunov function is selected that weighs only the tracking error terms to demonstrate the closed-loop stability and tracking performance. By contrast, in our proof, the Lyapunov function shown in the Appendix weighs the tracking errors $e(k)$ and the weight estimation errors of all the NNs for the controller $\tilde{W}(k)$. The proof is exceedingly complex due to the presence of several different variables. However, it obviates the need for the CE assumption, and it allows weight-tuning algorithms to be derived during the proof, not selected *a priori* in an ad hoc manner.

Remark 3: In this paper, two-layer NNs are utilized as OLAs for action and critic network signals whereas any other OLAs, such as CMAC, splines, fuzzy logic, and so on, can be utilized instead. Lyapunov proof of the controller convergence still holds.

Next, the output-feedback controller is discussed.

IV. Output-Feedback Online Reinforcement Learning Controller Design

In the state-feedback design that is presented in the previous section, all states are assumed to be available for the controller. However, in this section, the output-feedback version of our online reinforcement learning scheme is introduced when certain states of the plant are unavailable. First, an observer design is introduced followed by the action and critic NN designs before discussing the stability of the closed-loop system with output feedback.

A. Observer Structure

Consider system (1) and assume that only the output vector $y(k) \in \Re^m$ is available at the kth step. Therefore, to estimate other system states, an NN observer is first proposed. For the system described by (1), we use the following NN-based state observer to estimate the actual state $x(k)$ as

$$\begin{aligned} \hat{x}_1(k) &= \hat{x}_2(k-1) \\ &\vdots \\ \hat{x}_n(k) &= \hat{w}_o^T(k-1)\phi_o\left(v_o^T \hat{z}(k-1)\right) \\ &= \hat{w}_o^T(k-1)\phi_o\left(\hat{z}(k-1)\right) \end{aligned} \tag{39}$$

where $\hat{x}_i(k) \in \Re^m$, $i = 1, \ldots, n$, is the estimated value of $x_i(k) \in \Re^m$, $\hat{z}(k-1) = [\hat{x}_1^T(k-1), \ldots, \hat{x}_n^T(k-1), u^T(k-1)]^T \in \Re^{(n+1)m}$ is the input vector to the NN observer at the kth instant, $\hat{w}_o(k-1) \in \Re^{n_o \times m}$ and $v_o \in \Re^{(n+1)m \times n_o}$ denote the output and hidden layer weights of the NN observer, and n_o is the number of the hidden layer neurons. For simplicity purposes, the hidden layer activation function vector $\phi_o(v_o^T \hat{z}(k-1)) \in \Re^{n_o}$ is written as $\phi_o(\hat{z}(k-1))$.

Define the state estimation error as

$$\tilde{x}_i(k) = \hat{x}_i(k) - x_i(k), \qquad i = 1, \ldots, n \tag{40}$$

where $\tilde{x}_i(k) \in \Re^m$, $i = 1, \ldots, n$, is the state estimation error. As a matter of fact, by comparing (1) and (39), one can find that the observer NN approximates the nonlinear function given by $f(x(k-1)) + g(x(k-1))u(k-1)$. Thus, ideally, this nonlinear function can be expressed as

$$\begin{aligned} &f\left(x(k-1)\right) + g\left(x(k-1)\right)u(k-1) \\ &\quad = w_o^T \phi_o\left(v_o^T z(k-1)\right) + \varepsilon_o\left(z(k-1)\right) \\ &\quad = w_o^T \phi_o\left(z(k-1)\right) + \varepsilon_o\left(z(k-1)\right) \end{aligned} \tag{41}$$

where $w_o \in \Re^{n_o \times m}$ is the target observer NN weight matrix, $\varepsilon_o(z(k-1))$ is the NN approximation error, and the NN input is $z(k-1) = [x_1^T(k-1), \ldots, x_n^T(k-1), u^T(k-1)]^T \in \Re^{(n+1)m}$. Again, for convenience, the hidden layer activation function vector $\phi_o(v_o^T z(k-1)) \in \Re^{n_o}$ is written as $\phi_o(z(k-1))$.

Combining (39)–(41), one obtains

$$\begin{aligned} \tilde{x}_n(k) &= \hat{x}_n(k) - x_n(k) \\ &= \hat{x}_n(k) - f\left(x(k-1)\right) \\ &\quad - g\left(x(k-1)\right)u(k-1) - d(k-1) \\ &= \hat{w}_o^T(k-1)\phi_o\left(\hat{z}(k-1)\right) - w_o^T\phi_o\left(z(k-1)\right) \\ &\quad - \varepsilon_o\left(z(k-1)\right) - d(k-1) \\ &= \left(\hat{w}_o^T(k-1) - w_o^T\right)\phi_o\left(\hat{z}(k-1)\right) \\ &\quad + w_o^T\left(\phi_o\left(\hat{z}(k-1)\right) - \phi_o\left(z(k-1)\right)\right) \\ &\quad - \varepsilon_o\left(z(k-1)\right) - d(k-1) \\ &= \tilde{w}_o^T(k-1)\phi_o\left(\hat{z}(k-1)\right) + w_o^T\phi_o\left(\tilde{z}(k-1)\right) \\ &\quad - \varepsilon_o\left(z(k-1)\right) - d(k-1) \\ &= \xi_o(k-1) + d_o(k-1) \end{aligned} \tag{42}$$

where

$$\tilde{w}_o(k-1) = \hat{w}_o(k-1) - w_o \tag{43}$$

$$\xi_o(k-1) = \tilde{w}_o^T(k-1)\phi_o\left(\hat{z}(k-1)\right) \tag{44}$$

$$\tilde{\phi}_o\left(z(k-1)\right) = \phi_o\left(\hat{z}(k-1)\right) - \phi_o\left(z(k-1)\right) \tag{45}$$

$$d_o(k-1) = w_o^T\tilde{\phi}_o\left(z(k-1)\right) - \varepsilon_o\left(z(k-1) + d(k-1).\right. \tag{46}$$

Therefore, the dynamics of the estimation error is obtained using (40) and (42) as

$$\begin{aligned} \tilde{x}_1(k) &= \tilde{x}_2(k-1) \\ &\vdots \\ \tilde{x}_n(k) &= \xi_o(k-1) + d_o(k-1). \end{aligned} \tag{47}$$

B. Action and Critic Network Designs

Since the actual system states are unavailable for the action and critic NNs, their input and update laws have to be changed accordingly. The basic idea is to substitute the unavailable system states with the corresponding estimated values from the observer NN. Consequently, the action NN input is taken as $\hat{s}(k) = [\hat{x}^T(k), y_d^T(k), y_d^T(k+n)]^T \in \Re^{(n+2)m}$, while the input to the critic NN is replaced by $\hat{x}(k)$. Thus, in our output-feedback design, the control input to the plant is provided as

$$u(k) = \hat{w}_a^T(k)\phi_a\left(v_a^T\hat{s}(k)\right) = \hat{w}_a^T(k)\phi_a\left(\hat{s}(k)\right). \tag{48}$$

The long-term cost function is approximated by

$$\hat{J}(k) = \hat{w}_c^T(k)\phi_c\left(v_c^T\hat{x}(k)\right) = \hat{w}_c^T(k)\phi_c\left(\hat{x}(k)\right). \tag{49}$$

Meanwhile, the training algorithms governing action and critic NNs are updated as

$$\begin{aligned}\hat{w}_a(k+1) =& \hat{w}_a(k) - \alpha_a\phi_a\left(\hat{s}(k)\right)\\ &\times\left(\hat{e}_n(k+1) - l_1\hat{e}_n(k) + \bar{J}(k)\right)^T \end{aligned}\tag{50}$$

$$\begin{aligned}\hat{w}_c(k+1) =& \hat{w}_c(k) - \alpha_c\gamma\phi_c\left(\hat{x}(k)\right)\\ &\times\left(\gamma\hat{J}(k) + r(k) - \hat{J}(k-1)\right)\end{aligned}\tag{51}$$

where $\hat{e}_n(k) = \hat{x}_n(k) - x_{nd}(k)$.

Thus, (10) has to be rewritten as

$$\begin{aligned}e_n(k+1) =& f\left(x(k)\right) + g\left(x(k)\right)u(k) + d(k) - y_d(k+n)\\ =& l_1e_n(k) + g\left(x(k)\right)\left(u(k) - u_d(k)\right) + d(k)\\ =& l_1e_n(k) + g\left(x(k)\right)\\ &\times\left(\tilde{w}_a^T(k)\phi_a\left(\hat{s}(k)\right) - \varepsilon_a\left(\hat{s}(k)\right)\right) + d(k)\\ =& l_1e_n(k) + g\left(x(k)\right)\hat{\zeta}_a(k) + \hat{d}_a(k)\end{aligned}\tag{52}$$

where

$$\hat{\zeta}_a(k) = \tilde{w}_a^T(k)\phi_a\left(\hat{s}(k)\right) \tag{53}$$

$$\hat{d}_a(k) = -g\left(x(k)\right)\varepsilon_a\left(\hat{s}(k)\right) + d(k). \tag{54}$$

C. Weight Updates for the Observer NN

The observer NN weight update is driven by the state estimation error $\tilde{x}_1(k) = \hat{x}_1(k) - y(k)$, i.e.,

$$\begin{aligned}\hat{w}_o(k+1) =& \hat{w}_o(k) - \alpha_o\phi_o\\ &\times\left(\hat{z}(k)\right)\left(\hat{w}_o^T(k)\phi_o(\hat{z}(k)) + l_2\tilde{x}_1(k)\right)^T\end{aligned}\tag{55}$$

where $l_2 \in \Re^{m\times m}$ is a design matrix and $\alpha_o \in \Re^+$ is the adaptation gain for the NN observer.

D. Theoretic Result

Theorem 2: Consider the system given by (1) with only the output vector available. Let the Assumptions 1–5 hold (Assumptions 4 and 5 also hold for the observer NN) with the disturbance bound d_m being a known constant. Let the system states be estimated by observer NN (39) and the control input be provided by the action NN (48), with the critic NN (49) tuning the action NN weights. Furthermore, let the observer, action, and critic NN weights be tuned by (50), (51) and (55), , respectively. Then, there exist four constants B_e, B_o, B_a, and B_c, denoted as the uniform ultimate bounds for the tracking error $e(k)$, and the NN weight estimates the errors of the observer, action, and critic NNs $\zeta_o(k)$, $\hat{\zeta}_a(k)$, and $\zeta_c(k)$, respectively, given by

$$B_e = \sqrt{3}D_M / \sqrt{\eta_4 - 3(\eta_3 + \eta_4)l_{1\max}^2} \tag{56}$$

$$B_o = D_M / \sqrt{\eta_6 - \eta_2 - 2\eta_7'} \tag{57}$$

$$B_a = D_M / \sqrt{\eta_7' g_{\min} - (\eta_3 + \eta_4)g_{\max}^2 - \eta_9} \tag{58}$$

$$B_c = D_M / \sqrt{\eta_8\gamma^2 - \eta_7' - \eta_{10}} \tag{59}$$

provided that the following additional conditions are met as

$$0 < \alpha_o\left\|\phi_o(k)\right\|^2 < 1 \tag{60}$$

$$\gamma > \frac{\sqrt{3}}{3}. \tag{61}$$

Proof: See the Appendix.

Remark 4: The observer NN weights can also be initialized at zero or random, which avoids the need for offline learning phase.

Remark 5: Since separation principle does not hold for nonlinear systems, the proposed output-feedback controller relaxes this strong assumption since Lyapunov proof includes observer estimation error and weight estimation error terms along with the action and critic network terms.

V. Simulation Results

To demonstrate the feasibility of the theoretic results, the online reinforcement learning controller design is implemented on a pendulum balancing system and a two-link robotic arm system by simulation.

A. Pendulum Balancing System

First, our approach is examined on a pendulum swing up and balancing task. The example under investigation is identical to that in [6] and [7]. The continuous-time dynamics of the pendulum can be written as follows:

$$\begin{aligned}\frac{d\omega}{dt} &= \frac{3}{4ml^2}(F + m\lg\sin\theta) + d\\ \frac{d\theta}{dt} &= \omega\end{aligned}\tag{62}$$

where $m = 1/3$ and $l = 3/2$ are the mass and length of the pendulum, respectively. The gravity $g = 0.98$ m/s^2. The original system states include the current angle θ and angular velocity ω, while the angular acceleration F is adopted as the control input. Furthermore, an additive bounded uniformly distributed noise $d \in [-0.02 \quad 0.02]$ is introduced with bound $d_m = 0.02$. The objective is to swing up the bar and to balance it at the top position, namely, $\theta_d = 0$, $\omega_d = 0$. Initially, the pendulum starts at $\theta = \pi$, which means that the bar is released loosely straight down.

In the implementation, the system dynamics are discretized with standard zero-order-hold technique presented in [5]. The time step is taken as 0.05. Since the optimal control law is unknown, for comparison purposes, the proposed reinforcement learning design and a traditional proportional–integral (PI) controller are evaluated.

First, the parameters of the online reinforcement learning design and the simulation parameters are selected according to Theorems 1 and 2 (Table I).

Traditional sigmoidal activation functions are employed in all the nodes in the hidden layer of all NNs [26]. The hidden layer weights are selected initially at random and held constant. The output layer weights are initialized randomly.

Then, the PI controller is implemented as $u = k_pr_\theta + k_I\int r_\theta dt$, where $r_\theta = \omega + k_\theta\theta$ is the filtered tracking error [26] and some of the design parameters are set as $k_\theta = 5$, $k_p = -5$,

TABLE I
SUMMARY OF PARAMETERS USED IN SIMULATION OF PENDULUM

Parameter	R	Q	γ	l_1	l_2	α_o
Value	diag{0.01,0.01}	0.1	0.5	0.1	0.5	0.8
Parameter	α_c	α_a	n_o	n_a	n_c	d_m
Value	0.01	0.1	5	10	10	0.02

TABLE II
SUMMARY OF PARAMETERS USED IN SIMULATION OF TWO-LINK ROBOT ARM

Parameter	m_1	m_2	a_1	a_2	n_c	R
Value	0.8	2.3	1	1	10	diag{2, 2, 2, 2}
Parameter	γ	d_m	l_1	l_2	n_o	Q
Value	0.5	0.1	0.2	0.5	10	diag{0.1, 0.1}
Parameter	n_a	α_o	α_c	α_a		
Value	40	0.01	1×10^{-4}	5×10^{-2}		

and $k_I = -20$. The PI controller parameters are obtained from numerous simulation tests but may not be globally "optimal."

In our study, 100 consecutive trials for state-feedback reinforcement learning design are attempted, and the task is successfully accomplished for every trial. Fig. 3(a) shows the simulation results of state-feedback controller for one of the trials in terms of θ and ω, and Fig. 3(b) gives the corresponding control signal. Consistent with the theoretical results, the tracking error converges to a small value, and the control input is also found to be bounded.

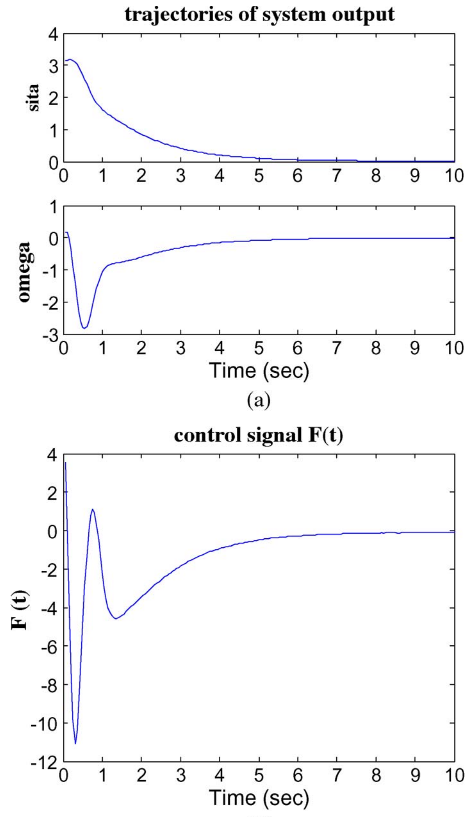
Meanwhile, the performance of the PI controller is shown in Fig. 4. It is apparent that, with PI controller, the system output trajectories converge to zero faster and seem to have a better performance. However, it should be noted that a lot more control effort is generated for PI control law by looking into the control input plots. As a matter of fact, the long-term cost function computed from (2) and (3) with the same R and Q is 9.981 for the online learning scheme but 13.731 for the PI controller, which shows that the proposed design is indeed tuned to decrease the cost function as predicted.

In addition, the time histories of both the critic NN weights and Bellman error (15) are demonstrated in Fig. 5. Examining the figure shows that the parameter estimates converge to constant values and remain bounded consistent with Theorem 1. Moreover, in order to show the impact of the cost function parameters on the controller performance, the simulation is redone with different settings of $R = \mathrm{diag}\{0.05, 0.05\}$ and $Q = 0.5$. As one can see in Fig. 6, the NN weights are still bounded, and Bellman error converges to zero eventually as expected.

Furthermore, Fig. 7 shows the output trajectories of the system with output-feedback online learning controller, which verifies its feasibility.

B. Two-Link Planar Robot Arm System

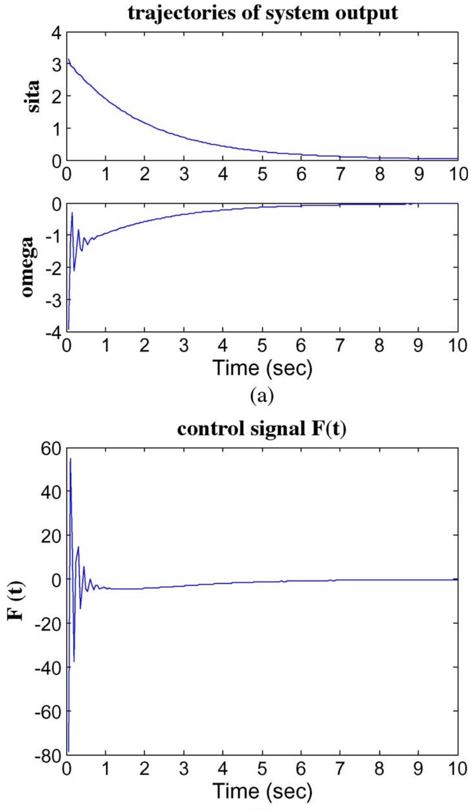
In the second implementation, the two-link planar robot arm system shown in Fig. 8 and discussed in [5] is considered.



(a)

(b)

Fig. 3. Performance of the state-feedback online reinforcement learning controller. (a) Trajectories of system outputs θ and ω. (b) Control input.



(a)

(b)

Fig. 4. Simulation results of the PI controller on pendulum balancing system. (a) Trajectories of system outputs θ and ω. (b) Control input.

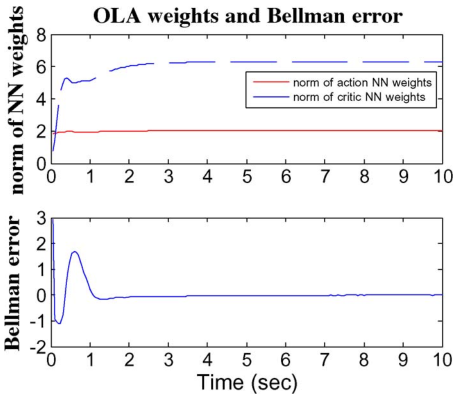


Fig. 5. Trajectories of OLA weights and Bellman error.

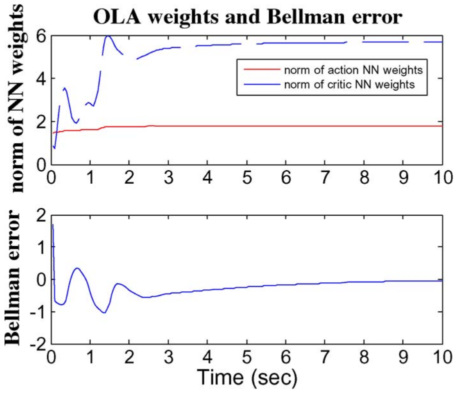


Fig. 6. Trajectories of OLA weights and Bellman error along with $R = \text{diag}\{0.05, 0.05\}$ and $Q = 0.5$.

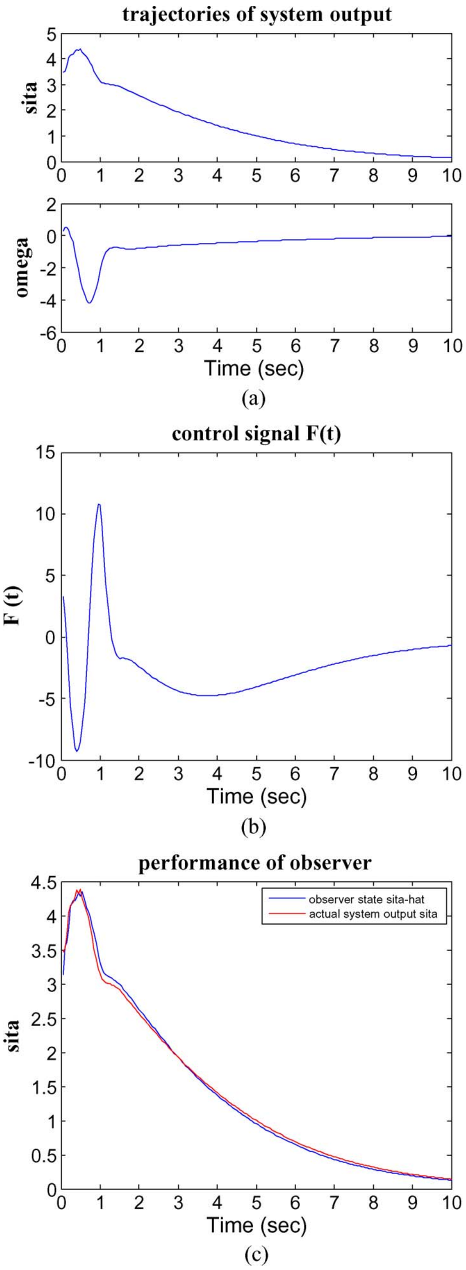
The continuous-time manipulator dynamics is as follows [5]:

$$\begin{bmatrix} \alpha+\beta+2\eta\cos q_2 & \beta+\eta\cos q_2 \\ \beta+\eta\cos q_2 & \beta \end{bmatrix}\begin{bmatrix}\ddot{q}_1 \\ \ddot{q}_2\end{bmatrix} + \begin{bmatrix} -\eta\left(2\dot{q}_1\dot{q}_2+\dot{q}_2^2\right)\sin q_2 \\ \eta\dot{q}_1^2\sin q_2 \end{bmatrix} + \begin{bmatrix} \alpha e_1\cos q_1 + \eta e_1\cos(q_1+q_2) \\ \eta e_1\cos(q_1+q_2)\end{bmatrix} + \begin{bmatrix} d_1 \\ d_2\end{bmatrix} = \begin{bmatrix}\tau_1 \\ \tau_2\end{bmatrix} \quad (63)$$

where

α $= (m_1+m_2)a_1^2$;
β $= m_2 a_2^2$;
η $= m_2 a_1 a_2$;
e_1 $= g/a_1$;
g $= 9.8$ m/s^2, the acceleration of gravity;
m_1, m_2 point masses of the links at distal end;
a_1, a_2 lengths of the links;
q_1, q_2 rotational angles of the joints;
τ_1, τ_2 torques applied on the joints.

In most of the controller designs, joint angles q_1 and q_2 are the states while τ_1 and τ_2 are the control inputs. After using the same technique as that of the pendulum example for discretization, the system dynamics in discrete time can be written in an affine form as (1). In the simulation, the time step is set as 1 ms. To be more realistic, the system is also added with a bounded random disturbance $d = [d_1 \quad d_2]^T$, where d_1, $d_2 \in [-0.1 \quad 0.1]$. The initial states of the system are set at $q_1(0) = q_2(0) = 0$, and for the output-feedback case, the initial states of the observer are $\hat{q}_1(0) = \hat{q}_2(0) = 0$ as well. Our goal



(a)

(b)

(c)

Fig. 7. Simulation results of the output-feedback online reinforcement learning controller on pendulum balancing system. (a) Trajectories of system outputs θ and ω. (b) Control input. (c) Comparison of observer states and actual system outputs.

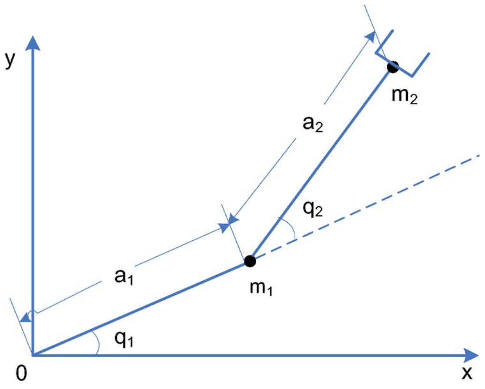
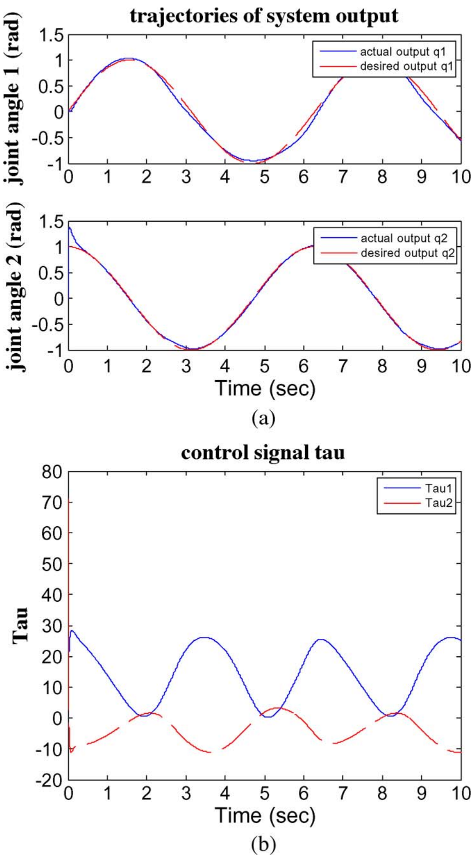


Fig. 8. Two-link planar robot arm.



(a)

(b)

Fig. 9. Simulation results of the state-feedback online reinforcement learning controller on two-link planar robot arm. (a) (Solid line) Trajectories of the actual rotational angles; (Dashed line) Desired final values of the angles. (b) Control input signals τ_1 and τ_2.

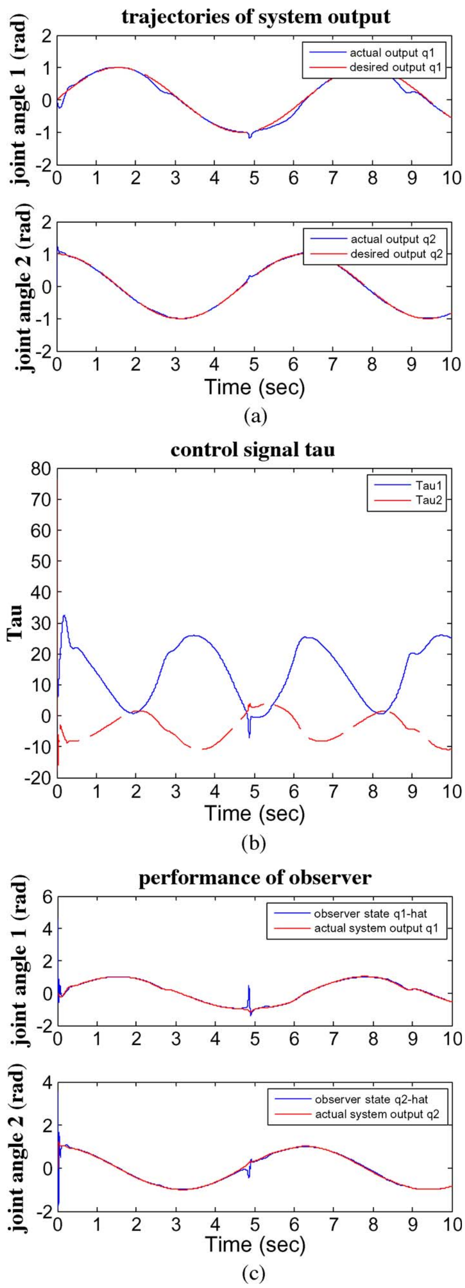
is to manipulate the robot arm to track a sinusoidal signal with a frequency of $1/2\pi$ and an amplitude of 1 rad. The simulation parameters used in this simulation are tabulated in Table II.

A typical system response using state-feedback-based online reinforcement learning controller is shown in Fig. 9 including the system output trajectories and control signals. At the same time, Fig. 10 shows the system response with its output-feedback counterpart in terms of the system output trajectories, control signals, and, also, the observer states. From the simulation results, we can find that both the designs are capable of accomplishing the control targets with all signals of the closed-loop system bounded.

In order to verify the robustness of the output-feedback control design with respect to the observer's initial conditions, the initial states of the observer are intentionally set as $\hat{q}_1(0) = \hat{q}_2(0) = \pi/2$. Fig. 11 shows the state trajectories, control signals, and the actual and the estimated system states from the NN observer, which also confirms that the proposed controller is able to maintain the tracking performance even with initial conditions.

VI. Conclusion

A novel reinforcement-learning-based online neural controller has been designed for affine nonlinear systems to deliver a desired performance under bounded disturbance. Both state-feedback and output-feedback versions are introduced in this paper. The proposed NN controller optimizes the long-term cost function by introducing a critic NN. Unlike the many applications where the controller is trained offline, the control input is updated in an online fashion. To guarantee that a control system must be stable all of the time, the boundedness of the closed-loop tracking errors and NN weight estimates is verified by using Lyapunov analysis in the presence of bounded disturbances and approximation errors. The observer estimates unavailable system states in the output-feedback design. Persistency of excitation condition, CE, and separation principles are not required. The feasibility of the two methods is also strengthened through the controller implementations on pendulum and two-link robotic arm.

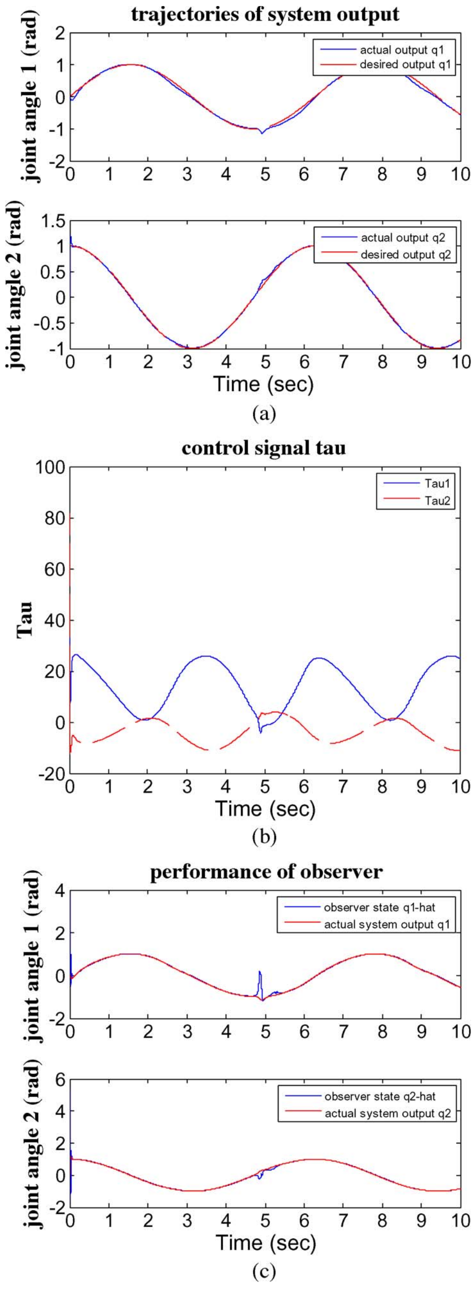


(a)

(b)

(c)

Fig. 10. Performance of the output-feedback online reinforcement learning controller with initial observer state $\hat{q}_1(0) = \hat{q}_2(0) = 0$. (a) (Solid line) Trajectories of the actual rotational angles; (Dashed line) Desired final values of the angles. (b) Control input signals τ_1 and τ_2. (c) Comparison of observer states and actual system outputs.



(a)

(b)

(c)

Fig. 11. Performance of the output-feedback online reinforcement learning controller with initial observer state $\hat{q}_1(0) = \hat{q}_2(0) = \pi/2$. (a) (Solid line) Trajectories of the actual rotational angles; (Dashed line) Desired final values of the angles. (b) Control input signals τ_1 and τ_2. (c) Comparison of observer states and actual system outputs.

APPENDIX

Proof of Theorem 1: Consider the following Lyapunov candidate

$$\begin{aligned} L_s(k) &= \sum_{i=1}^{4} L_{si} \\ &= \frac{\gamma_1}{3}\|e_n(k)\|^2 + \frac{\gamma_2}{\alpha_a} tr\left(\tilde{W}_a^T(k)\tilde{W}_a(k)\right) \\ &\quad + \frac{\gamma_3}{\alpha_c} tr\left(\tilde{W}_c^T(k)\tilde{W}_c(k)\right) + \gamma_4\|\zeta_c(k-1)\|^2 \end{aligned} \tag{A.1}$$

where $\gamma_i \in \Re^+$, $i = 1, 2, 3, 4$, represents the design parameters. Hence, the first difference of the Lyapunov function is given by

$$\begin{aligned} \Delta L_{s1} &= \frac{\gamma_1}{3}\Big(\|e_n(k+1)\|^2 - \|e_n(k)\|^2\Big) \\ &= \frac{\gamma_1}{3}\Big(\|le_n(k) + g\left(x(k)\right)\zeta_a(k) + d_a(k)\|^2 - \|e_n(k)\|^2\Big) \\ &\le -\frac{\gamma_1}{3}\left(1 - 3l_{\max}^2\right)\|e_n(k)\|^2 \\ &\quad + \gamma_1 g_{\max}^2\|\zeta_a(k)\|^2 + \gamma_1\|d_a(k)\|^2 \end{aligned} \tag{A.2}$$

$$\begin{aligned} \Delta L_{s2} &= \frac{\gamma_2}{\alpha_a} tr\left(\tilde{W}_a^T(k+1)\tilde{W}_a(k+1) - \tilde{W}_a^T(k)\tilde{W}_a(k)\right) \\ &= \frac{\gamma_2}{\alpha_a} tr\Big(-2\tilde{W}_a^T(k)\alpha_a\phi_a\left(s(k)\right) \\ &\qquad \times \left(g\left(x(k)\right)\zeta_a(k) + \bar{J}(k) + d_a(k)\right) \\ &\qquad + \Delta\hat{W}_a^T(k)\Delta\hat{W}_a(k)\Big) \\ &= -2\gamma_2\zeta_a^T(k)g(x(k))\zeta_a(k) - 2\gamma_2\zeta_a^T(k)\left(\bar{J}(k) + d_a(k)\right) \\ &\quad + \gamma_2\alpha_a\|\phi_a\left(s(k)\right)\|^2\left\|g\left(x(k)\right)\zeta_a(k) + \bar{J}(k) + d_a(k)\right\|^2 \\ &\le -2\gamma_2 g_{\min}\|\zeta_a(k)\|^2 - 2\gamma_2\zeta_a^T(k)\left(\bar{J}(k) + d_a(k)\right) \\ &\quad + \gamma_2\alpha_a\|\phi_a\left(s(k)\right)\|^2 g_{\max}^2\|\zeta_a(k)\|^2 \\ &\quad + \gamma_2\alpha_a\|\phi_a\left(s(k)\right)\|^2 \\ &\quad \times \left(\left\|\bar{J}(k) + d_a(k)\right\|^2 + 2\left(\bar{J}(k) + d_a(k)\right)^T g(x(k))\zeta_a(k)\right) \\ &= \gamma_2\Big\{-g_{\min}\|\zeta_a(k)\|^2 - \left(g_{\min} - \alpha_a\|\phi_a\left(s(k)\right)\|^2 g_{\max}^2\right) \\ &\qquad \times \|\zeta_a(k)\|^2 - 2\zeta_a^T(k) \\ &\qquad \times \left(I - \alpha_a\|\phi_a\left(s(k)\right)\|^2 g\left(x(k)\right)\right)\left(\bar{J}(k) + d_a(k)\right) \\ &\qquad + \alpha_a\|\phi_a\left(s(k)\right)\|^2\left\|\bar{J}(k) + d_a(k)\right\|^2\Big\} \\ &= \gamma_2\Bigg\{-g_{\min}\|\zeta_a(k)\|^2 + \frac{1 - \alpha_a\|\phi_a\left(s(k)\right)\|^2 g_{\min}}{g_{\min} - \alpha_a\|\phi_a\left(s(k)\right)\|^2 g_{\max}^2} \\ &\qquad \times \left\|\bar{J}(k) + d_a(k)\right\|^2 \\ &\qquad - \left(g_{\min} - \alpha_a\|\phi_a\left(s(k)\right)\|^2 g_{\max}^2\right) \\ &\qquad \times \left\|\zeta_a(k) + \frac{\left(I - \alpha_a\|\phi_a\left(s(k)\right)\|^2 g\left(x(k)\right)\right)}{g_{\min} - \alpha_a\|\phi_a(s(k))\|^2 g_{\max}^2}\right\|^2\Bigg\}. \end{aligned} \tag{A.3}$$

Set $\gamma_2 = \gamma_2'\gamma_2''$, where $\gamma_2''(1 - \alpha_a\|\phi_a(s(k))\|^2 g_{\min}/g_{\min} - \alpha_a\|\phi_a(s(k))\|^2 g_{\max}^2) \le (1/2)$; hence

$$\begin{aligned} \Delta L_{s2} &\le -\gamma_2 g_{\min}\|\zeta_a(k)\|^2 - \gamma_2\left(g_{\min} - \alpha_a\|\phi_a(s(k))\|^2 g_{\max}^2\right) \\ &\quad \times \left\|\zeta_a(k) + \frac{\left(I - \alpha_a\|\phi_a\left(s(k)\right)\|^2 g\left(x(k)\right)\right)}{g_{\min} - \alpha_a\|\phi_a\left(s(k)\right)\|^2 g_{\max}^2}\right\|^2 \\ &\quad + \gamma_2'\frac{\left\|\bar{J}(k) + d_a(k)\right\|^2}{2} \\ &\le -\gamma_2 g_{\min}\|\zeta_a(k)\|^2 - \gamma_2\left(g_{\min} - \alpha_a\|\phi_a(s(k))\|^2 g_{\max}^2\right) \\ &\quad \times \left\|\zeta_a(k) + \frac{\left(I - \alpha_a\|\phi_a\left(s(k)\right)\|^2 g\left(x(k)\right)\right)}{g_{\min} - \alpha_a\|\phi_a\left(s(k)\right)\|^2 g_{\max}^2}\right\|^2 \\ &\quad + \gamma_2' n\|\zeta_c(k)\|^2 + \gamma_2' n\|J^*(k) + d_a(k)\|^2. \end{aligned} \tag{A.4}$$

At the same time

$$\Delta L_{s3} = \frac{\gamma_3}{\alpha_c} tr\left(\tilde{W}_c^T(k+1)\tilde{W}_c(k+1) - \tilde{W}_c^T(k)\tilde{W}_c(k)\right)$$

$$\begin{aligned}
&= \frac{\gamma_3}{\alpha_c} tr\left(-2\tilde{W}_c^T(k)\alpha_c\gamma\phi_c\left(x(k)\right)e_c(k)\right)\\
&\quad + \frac{\gamma_3}{\alpha_c} tr\left(\Delta\hat{W}_c^T(k)\Delta\hat{W}_c(k)\right)\\
&= -2\gamma_3\gamma\zeta_c(k)e_c(k) + \gamma_3\alpha_c\gamma^2 e_c^2(k)\left\|\phi_c(k)\right\|^2\\
&\leq -\gamma_3\left(1-\alpha_c\gamma^2\left\|\phi_c(k)\right\|^2\right)e_c^2(k) - \gamma_3\gamma^2\zeta_c^2(k)\\
&\quad + \frac{\gamma_3}{4}\zeta_c^2(k-1) + \frac{\gamma_3}{4}\left(\gamma J^*(k) - J^*(k-1)\right)^2\\
&\quad + \frac{\gamma_3}{4}r(k) + \frac{\gamma_3}{4}\left(\varepsilon_c(k) - \varepsilon_c(k-1)\right)^2\\
&\leq -\gamma_3\left(1-\alpha_c\gamma^2\left\|\phi_c(k)\right\|^2\right)e_c^2(k) - \gamma_3\gamma^2\zeta_c^2(k)\\
&\quad + \frac{\gamma_3}{4}\zeta_c^2(k-1) + \frac{\gamma_3}{4}\left(\gamma J^*(k) - J^*(k-1)\right)^2\\
&\quad + \frac{\gamma_3}{4}Q_{\max}\left\|e(k)\right\|^2 + \frac{\gamma_3}{4}\left(\zeta_a(k) + w_a^T\phi_a(k)\right)^T R\\
&\quad \times\left(\zeta_a(k) + w_a^T\phi_a(k)\right) + \gamma_3\varepsilon_{cm}^2\\
&\leq -\gamma_3\left(1-\alpha_c\gamma^2\left\|\phi_c(k)\right\|^2\right)e_c^2(k) - \gamma_3\gamma^2\zeta_c^2(k)\\
&\quad + \frac{\gamma_3}{4}\zeta_c^2(k-1) + \frac{\gamma_3}{4}\left(\gamma J^*(k) - J^*(k-1)\right)^2\\
&\quad + \frac{\gamma_3}{4}Q_{\max}\left\|e(k)\right\|^2 + \frac{\gamma_3}{8}R_{\max}\left\|\zeta_a(k)\right\|^2\\
&\quad + \frac{\gamma_3}{8}R_{\max}\left\|w_a^T\phi_a(k)\right\|^2 + \gamma_3\varepsilon_{cm}^2
\end{aligned}\tag{A.5}$$

where $Q_{\max}$ and $R_{\max}$ are the maximum eigenvalues of matrices Q and R, respectively

$$\Delta L_{s4} = \gamma_4\left(\left\|\zeta_c(k)\right\|^2 - \left\|\zeta_c(k-1)\right\|^2\right).\tag{A.6}$$

Combining (A.1)–(A.6) yields

$$\begin{aligned}
\Delta L_s(k) \leq& -\frac{\gamma_1}{3}\left(1-3l_{\max}^2\right)\left\|e_n(k)\right\|^2 + \gamma_1 g_{\max}^2\left\|\zeta_a(k)\right\|^2\\
&+ \gamma_1\left\|d_a(k)\right\|^2 - \gamma_2 g_{\min}\left\|\zeta_a(k)\right\|^2\\
&- \gamma_2\left(g_{\min} - \alpha_a\left\|\phi_a\left(s(k)\right)\right\|^2 g_{\max}^2\right)\\
&\times\left\|\zeta_a(k) + \frac{\left(I - \alpha_a\left\|\phi_a\left(s(k)\right)\right\|^2 g\left(x(k)\right)\right)}{g_{\min} - \alpha_a\left\|\phi_a\left(s(k)\right)\right\|^2 g_{\max}^2}\right\|^2\\
&+ \gamma_2' n\left\|\zeta_c(k)\right\|^2 + \gamma_2' n\left\|J^*(k) + d_a(k)\right\|^2\\
&- \gamma_3\left(1-\alpha_c\gamma^2\left\|\phi_c(k)\right\|^2\right)e_c^2(k)\\
&- \gamma_3\gamma^2\zeta_c^2(k) + \frac{\gamma_3}{4}\zeta_c^2(k-1) + \frac{\gamma_3}{4}\\
&\times\left(\gamma J^*(k) - J^*(k-1)\right)^2 + \frac{\gamma_3}{4}Q_{\max}\left\|e_n(k)\right\|^2\\
&+ \frac{\gamma_3}{8}R_{\max}\left\|\zeta_a(k)\right\|^2 + \frac{\gamma_3}{8}R_{\max}\left\|w_a^T\phi_a(k)\right\|^2\\
&+ \gamma_4\left(\left\|\zeta_c(k)\right\|^2 - \left\|\zeta_c(k-1)\right\|^2\right) + \gamma_3\varepsilon_{cm}^2\\
=& -\left(\frac{\gamma_1}{3}\left(1-3l_{\max}^2\right) - \frac{\gamma_3}{4}Q_{\max}\right)\left\|e_n(k)\right\|^2\\
&- \left(\gamma_2 g_{\min} - \gamma_1 g_{\max}^2 - \frac{\gamma_3}{8}R_{\max}\right)\left\|\zeta_a(k)\right\|^2\\
&- \left(\gamma_3\gamma^2 - \gamma_2' n - \gamma_4\right)\left\|\zeta_c(k)\right\|^2 - \left(\gamma_4 - \frac{\gamma_3}{4}\right)\\
&\times\left\|\zeta_c(k-1)\right\|^2 - \gamma_3\left(1-\alpha_c\gamma^2\left\|\phi_c(k)\right\|^2\right)e_c^2(k)\\
&- \gamma_2\left(g_{\min} - \alpha_a\left\|\phi_a\left(s(k)\right)\right\|^2 g_{\max}^2\right)\\
&\times\left\|\zeta_a(k) + \frac{\left(I - \alpha_a\left\|\phi_a\left(s(k)\right)\right\|^2 g\left(x(k)\right)\right)}{g_{\min} - \alpha_a\left\|\phi_a\left(s(k)\right)\right\|^2 g_{\max}^2}\right\|^2 + D_M^2
\end{aligned}\tag{A.7}$$

where

$$\begin{aligned}
D_M^2 =& \frac{\gamma_2' n}{2}d_a^2(k) + \left(\frac{\gamma_3}{4} + \frac{\gamma_2' n}{2}\right)J_m^2 + \frac{\gamma_3}{6}R_{\max}\\
&\times\left\|w_a^T\phi_a(k)\right\|^2 + \gamma_1\left\|d_a(k)\right\|^2 + \gamma_3\varepsilon_{cm}^2.
\end{aligned}\tag{A.8}$$

For the standard Lyapunov analysis, (A.7) and (A.8) imply that $\Delta L_s < 0$ as long as the conditions (35)–(38) are satisfied and the following holds:

$$\left\|e_n(k)\right\| > 2\sqrt{3}D_M\Big/\sqrt{4\gamma_1\left(1-3l_{\max}^2\right) - 3\gamma_3 Q_{\max}}\tag{A.9}$$

or

$$\left\|\zeta_a(k)\right\| > 2\sqrt{2}D_M\Big/\sqrt{8\gamma_2 g_{\min} - 8\gamma_1 g_{\max}^2 - \gamma_3 R_{\max}}\tag{A.10}$$

or

$$\left\|\zeta_c(k)\right\| > D_M\Big/\sqrt{\gamma_3\gamma^2 - \gamma_2' n - \gamma_4}.\tag{A.11}$$

According to the standard Lyapunov extension theorem [5], [26], the aforementioned analysis demonstrates that the tracking error $\|e_n(k)\|$ and the weights of the estimation errors are UUB. Considering $e_i(k) = x_i(k) - x_{id}(k) = e_n(k-n+i)$, $i = 1,\ldots,n$, one can readily conclude that $e_i(k)$, $i = 1,\ldots,n-1$, is also UUB. Furthermore, the boundedness of $\|\zeta_a(k)\|$ and $\|\zeta_c(k)\|$ implies that the weight estimations $\|\hat{w}_a(k)\|$ and $\|\hat{w}_c(k)\|$ are also bounded.

Proof of Theorem 2: The proof of Theorem 2 is similar to that of Theorem 1. Since an additional observer NN is introduced to estimate the immeasurable states, we consider the following Lyapunov function:

$$\begin{aligned}
L_o(k) =& \sum_{i=1}^{11} L_{oi} = \frac{\eta_1}{2}\sum_{i=1}^{n}\left\|\tilde{x}_i(k-1)\right\|^2 + \frac{\eta_2}{2}\sum_{i=1}^{n}\left\|\tilde{x}_i(k)\right\|^2\\
&+ \frac{\eta_3}{3}\sum_{i=1}^{n}\left\|e_i(k-1)\right\|^2 + \frac{\eta_4}{3}\sum_{i=1}^{n}\left\|e_i(k)\right\|^2 + \frac{\eta_5}{\alpha_o}\\
&\times tr\left(\tilde{W}_o^T(k-1)\tilde{W}_o(k-1)\right)\\
&+ \frac{\eta_6}{\alpha_o}tr\left(\tilde{W}_o^T(k)\tilde{W}_o(k)\right) + \frac{\eta_7}{\alpha_a}tr\left(\tilde{W}_a^T(k)\tilde{W}_a(k)\right)\\
&+ \frac{\eta_8}{\alpha_c}tr\left(\tilde{W}_c^T(k)\tilde{W}_c(k)\right) + \eta_9\left\|\zeta_c(k-1)\right\|^2\\
&+ \eta_{10}\left\|\zeta_c(k-1)\right\|^2 + \eta_{11}\left\|e_1(k-1)\right\|^2
\end{aligned}\tag{A.12}$$

where $\eta_i \in \Re^+$, $i = 1,\ldots,11$, represents the design parameters. Hence, the first difference of the Lyapunov function is the summation of the difference for each term.

From (42), we have

$$\begin{aligned}
\Delta L_{o1} =& \frac{\eta_1}{2}\sum_{i=1}^{n}\left\|\tilde{x}_i(k)\right\|^2 - \frac{\eta_1}{2}\sum_{i=1}^{n}\left\|\tilde{x}_i(k-1)\right\|^2\\
=& \eta_1\left\|\zeta_o(k-1)\right\|^2 + \eta_1\left\|d_o(k-1)\right\|^2 - \frac{\eta_1}{2}\left\|\tilde{x}_1(k-1)\right\|^2
\end{aligned}\tag{A.13}$$

$$\begin{aligned}
\Delta L_{o2} =& \frac{\eta_2}{2}\sum_{i=1}^{n}\left\|\tilde{x}_i(k+1)\right\|^2 - \frac{\eta_2}{2}\sum_{i=1}^{n}\left\|\tilde{x}_i(k)\right\|^2\\
=& \eta_2\left\|\zeta_o(k)\right\|^2 + \eta_2\left\|d_o(k)\right\|^2 - \frac{\eta_2}{2}\left\|\tilde{x}_1(k)\right\|^2.
\end{aligned}\tag{A.14}$$

Equation (52) yields

$$\begin{aligned}\Delta L_{o3} &= \frac{\eta_3}{3}\sum_{i=1}^{n}\|e_i(k+1)\|^2 - \frac{\eta_3}{3}\sum_{i=1}^{n}\|e_i(k)\|^2 \\ &\leq \eta_3 l_{1\max}^2 \|e_n(k)\|^2 + \eta_3 g_{\max}^2 \left\|\hat{\zeta}_a(k)\right\|^2 \\ &\quad + \eta_3 \left\|\hat{d}_a(k)\right\|^2 - \frac{\eta_3}{3}\|e_1(k)\|^2 \qquad \text{(A.15)} \\ \Delta L_{o4} &= \frac{\eta_4}{3}\|e_n(k+1)\|^2 - \frac{\eta_4}{3}\|e_n(k)\|^2 \\ &\leq \eta_4 l_{1\max}^2 \|e_n(k)\|^2 + \eta_4 g_{\max}^2 \left\|\hat{\zeta}_a(k)\right\|^2 \\ &\quad + \eta_4 \left\|\hat{d}_a(k)\right\|^2 - \frac{\eta_4}{3}\|e_n(k)\|^2. \qquad \text{(A.16)}\end{aligned}$$

Considering (55), we have

$$\begin{aligned}\Delta L_{o5} &= \frac{\eta_5}{\alpha_o} tr\left(\tilde{W}_o^T(k)\tilde{W}_o(k) - \tilde{W}_o^T(k-1)\tilde{W}_o(k-1)\right) \\ &\leq -\eta_5\left(1-\alpha_o\|\phi_o(k-1)\|^2\right) \\ &\quad \times \left\|\zeta_o(k-1) + l_2\tilde{x}_1(k-1) + W_o^T\phi_o(k-1)\right\|^2 \\ &\quad - \eta_5\|\zeta_o(k-1)\|^2 + 2\eta_5 l_{2\max}^2 \\ &\quad \times \|\tilde{x}_1(k-1)\|^2 + 2\eta_5 w_{om}^2\phi_{om}^2 \qquad \text{(A.17)} \\ \Delta L_{o6} &= \frac{\eta_6}{\alpha_o} tr\left(\tilde{W}_o^T(k+1)\tilde{W}_o(k+1) - \tilde{W}_o^T(k)\tilde{W}_o(k)\right) \\ &\leq -\eta_6\left(1-\alpha_o\|\phi_o(k)\|^2\right) \\ &\quad \times \left\|\zeta_o(k) + l_2\tilde{x}_1(k) + W_o^T\phi_o(k)\right\|^2 \\ &\quad - \eta_6\|\zeta_o(k)\|^2 + 2\eta_6 l_{2\max}^2\|\tilde{x}_1(k)\|^2 + 2\eta_6 w_{om}^2\phi_{om}^2 \\ &\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(A.18)}\end{aligned}$$

where $l_{2\max} \in \Re$ is the maximum eigenvalue of matrix l_2. Similar to (A.4), we obtain

$$\begin{aligned}\Delta L_{o7} &\leq -\eta_7'\left(g_{\min} - \alpha_a\|\phi_a(\hat{s}(k))\|^2 g_{\max}^2\right) \\ &\quad \times \left\|\hat{\zeta}_a(k) + \frac{\left(I - \alpha_a\|\phi_a(\hat{s}(k))\|^2 g(\hat{x}(k))\right)\beta(k)}{g_{\min} - \alpha_a\|\phi_a(\hat{s}(k))\|^2 g_{\max}^2}\right\|^2 \\ &\quad - \eta_7' g_{\min}\left\|\hat{\zeta}_a(k)\right\|^2 + \frac{\eta_7'}{4}\|\beta(k)\|^2 \\ &\leq -\eta_7'\left(g_{\min} - \alpha_a\|\phi_a(\hat{s}(k))\|^2 g_{\max}^2\right) \\ &\quad \times \left\|\hat{\zeta}_a(k) + \frac{\left(I - \alpha_a\|\phi_a(\hat{s}(k))\|^2 g(\hat{x}(k))\right)\beta(k)}{g_{\min} - \alpha_a\|\phi_a(\hat{s}(k))\|^2 g_{\max}^2}\right\|^2 \\ &\quad - \eta_7' g_{\min}\left\|\hat{\zeta}_a(k)\right\|^2 + 2\eta_7'\|\zeta_o(k)\|^2 \\ &\quad + 2\eta_7'\|d_o(k)\|^2 + 2\eta_7' l_{1\max}^2\|\zeta_o(k-1)\|^2 \\ &\quad + 2\eta_7' l_{1\max}^2\|d_o(k-1)\|^2 + \eta_7'\|\zeta_c(k)\|^2 \\ &\quad + \eta_7'\left\|W_c^T\phi_c(\hat{x}(k)) + \hat{d}_a(k)\right\|^2 \qquad \text{(A.19)}\end{aligned}$$

where $\beta(k) = \tilde{x}_n(k+1) - l_1\tilde{x}_n(k) + \hat{Q}(k) + \hat{d}_a(k)$ and $\eta_7 = \eta_7'\eta_7''$ such that $\eta_7''(1-\alpha_a\|\phi_a(\hat{s}(k))\|^2 g_{\min}/g_{\min} - \alpha_a\|\phi_a(\hat{s}(k))\|^2 g_{\max}^2) \leq (1/4)$. Similar to (A.5)

$$\begin{aligned}\Delta L_{o8} &\leq -\eta_8\left(1-\alpha_c\gamma^2\|\phi_c(\hat{x}(k))\|^2\right)e_c^2(k) \\ &\quad - \eta_8\gamma^2\zeta_c^2(k) + \frac{\eta_8}{3}\zeta_c^2(k-1) \\ &\quad + \frac{\eta_8}{3}Q_{\max}\|e_1(k-1)\|^2 \\ &\quad + \frac{\eta_8}{3}(\gamma J(k) - J(k-1))^2 + \frac{2\eta_8}{3}R_{\max}\left\|\hat{\zeta}_a(k-1)\right\|^2 \\ &\quad + \frac{2\eta_8}{3}R_{\max}\left\|w_a^T\phi_a(\hat{s}(k-1))\right\|^2. \qquad \text{(A.20)}\end{aligned}$$

Furthermore

$$\begin{aligned}\Delta L_{o9} &= \eta_9\left\|\hat{\zeta}_a(k)\right\|^2 - \eta_9\left\|\hat{\zeta}_a(k-1)\right\|^2 \qquad \text{(A.21)} \\ \Delta L_{o10} &= \eta_{10}\|\zeta_c(k)\|^2 - \eta_{10}\|\zeta_c(k-1)\|^2 \qquad \text{(A.22)} \\ \Delta L_{o11} &= \eta_{11}\|e_1(k)\|^2 - \eta_{11}\|e_1(k-1)\|^2. \qquad \text{(A.23)}\end{aligned}$$

Combining (A.13)–(A.23) yields

$$\begin{aligned}\Delta L_o \leq &-(\eta_6 - \eta_2 - 2\eta_7')\|\zeta_o(k)\|^2 \\ &- \left(\eta_5 - \eta_1 - 2\eta_7' l_{1\max}^2\right)\|\zeta_o(k-1)\|^2 \\ &- \left(\eta_7' g_{\min} - \eta_4 g_{\max}^2 - \eta_3 g_{\max}^2 - \eta_9\right)\left\|\hat{\zeta}_a(k)\right\|^2 \\ &- \left(\eta_9 - \frac{2\eta_8}{3}R_{\max}\right)\left\|\hat{\zeta}_a(k-1)\right\|^2 \\ &- \left(\eta_8\eta^2 - \eta_7' - \eta_{10}\right)\zeta_c^2(k) - \left(\eta_{10} - \frac{\eta_8}{3}\right)\zeta_c^2(k-1) \\ &- \left(\frac{\eta_2}{2} - 2\eta_6 l_{2\max}^2\right)\|\tilde{x}_1(k)\|^2 \\ &- \left(\frac{\eta_1}{2} - 2\eta_5 l_{2\max}^2\right)\|\tilde{x}_1(k-1)\|^2 - \left(\frac{\eta_3}{3} - \eta_{11}\right) \\ &\times \|e_1(k)\|^2 - \left(\eta_{11} - \frac{\eta_8}{3}Q_{\max}\right)\|e_1(k-1)\|^2 \\ &- \left(\frac{\eta_4}{3} - \eta_4 l_{1\max}^2 - \eta_3 l_{1\max}^2\right)\|e_n(k)\|^2 \\ &- \eta_8\left(1-\alpha_c\gamma^2\|\phi_c(\hat{x}(k))\|^2\right)e_c^2(k) \\ &- \eta_7'\left(g_{\min} - \alpha_a\|\phi_a(\hat{s}(k))\|^2 g_{\max}^2\right) \\ &\times \left\|\hat{\zeta}_a(k) + \frac{\left(I - \alpha_a\|\phi_a(\hat{s}(k))\|^2 g(\hat{x}(k))\right)\beta(k)}{g_{\min} - \alpha_a\|\phi_a(\hat{s}(k))\|^2 g_{\max}^2}\right\|^2 \\ &- \eta_6\left(1-\alpha_o\|\phi_o(k)\|^2\right)\left\|\zeta_o(k) + l_2\tilde{x}_1(k) + W_o^T\phi_o(k)\right\|^2 \\ &- \eta_5\left(1-\alpha_o\|\phi_o(k-1)\|^2\right) \\ &\times \left\|\zeta_o(k-1) + l_2\tilde{x}_1(k-1) + W_o^T\phi_o(k-1)\right\|^2 + D_M^2 \\ &\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(A.24)}\end{aligned}$$

where

$$\begin{aligned}D_M^2 &= (2\eta_5 + 2\eta_6)w_{om}^2\phi_{om}^2 + \frac{2\eta_8}{3}R_{\max}w_{am}^2\phi_{am}^2 \\ &\quad + 2\eta_7' w_{cm}^2\phi_{cm}^2 + \frac{\eta_8(2\gamma^2+1)}{3}J_m^2 \\ &\quad + \left(\eta_1 + \eta_2 + 2\eta_7' + 2\eta_7' l_{1\max}^2\right)d_{om}^2 + (\eta_3 + \eta_4 2\eta_7')\,d_{am}^2. \\ &\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(A.25)}\end{aligned}$$

Referring to the standard Lyapunov analysis [5], [26], (A.24) and (A.25) imply that $\Delta L_o < 0$ as long as the conditions (35)–(37) and (60)–(61) are satisfied and the following holds:

$$\|\tilde{x}_1(k)\| > \sqrt{2}D_M \Big/ \sqrt{\eta_2 - 4\eta_6 l_{2\max}^2} \qquad \text{(A.26)}$$

or

$$\|e_n(k)\| > \sqrt{3}D_M \Big/ \sqrt{\eta_4 - 3(\eta_3 + \eta_4)l_{1\max}^2} \qquad \text{(A.27)}$$

or

$$\|\zeta_o(k)\| > D_M \Big/ \sqrt{\eta_6 - \eta_2 - 2\eta_7'} \qquad \text{(A.28)}$$

or

$$\left\|\hat{\zeta}_a(k)\right\| > D_M \Big/ \sqrt{\eta_7' g_{\min} - (\eta_3 + \eta_4) g_{\max}^2 - \eta_9} \tag{A.29}$$

or

$$\|\zeta_c(k)\| > D_M \Big/ \sqrt{\eta_8 \gamma^2 - \eta_7' - \eta_{10}}. \tag{A.30}$$

According to the standard Lyapunov extension theorem [5], [26], the aforementioned analysis demonstrates that the tracking error $\|e(k)\|$ and the weights of the estimation errors are UUB. Furthermore, the boundedness of $\|\zeta_o(k)\|$, $\|\hat{\zeta}_a(k)\|$, and $\|\zeta_c(k)\|$ implies that the weight estimations $\|\hat{w}_o(k)\|$, $\|\hat{w}_a(k)\|$, and $\|\hat{w}_c(k)\|$ are also bounded.

REFERENCES

[1] R. Bellman and S. Dreyfus, *Applied Dynamic Programming*. Princeton, NJ: Princeton Univ. Press, 1962.

[2] Z. V. Rekasius, "Suboptimal design of intentionally nonlinear controllers," *IEEE Trans. Autom. Control*, vol. AC-9, no. 4, pp. 380–386, Oct. 1964.

[3] R. J. Leake and R. Liu, "Construction of suboptimal control sequences," *SIAM J. Control Optim.*, vol. 5, no. 1, pp. 54–63, 1967.

[4] D. Kirk, *Optimal Control Theory: An Introduction*. Englewood Cliffs, NJ: Prentice-Hall, 1970.

[5] F. L. Lewis, S. Jagannathan, and A. Yesildirek, *Neural Network Control of Robot Manipulators and Nonlinear Systems*. Philadelphia, PA: Taylor & Francis, 1999.

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

[7] J. C. Santamaria, R. S. Sutton, and A. Ram, "Experiments with reinforcement learning in problems with continuous state and action spaces," Univ. Massachusetts, Amherst, MA, COINS Tech. Rep. 96-88, Dec. 1996.

[8] R. Luus, *Iterative Dynamic Programming*. Boca Raton, FL: CRC Press, 2000.

[9] A. G. Barto, R. S. Sutton, and C. W. Anderson, "Neuron like adaptive elements that can solve difficult learning control problems," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-13, no. 5, pp. 834–847, Sep./Oct. 1983.

[10] R. S. Sutton, "Learning to predict by the methods of temporal difference," *Mach. Learn.*, vol. 3, no. 1, pp. 9–44, Aug. 1988.

[11] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, no. 3/4, pp. 279–292, May 1992.

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

[13] G. Boone, "Efficient reinforcement learning: Model-based acrobot control," in *Proc. IEEE Int. Conf. Robot. Autom.*, Albuquerque, NM, 1997, pp. 229–234.

[14] P. He and S. Jagannathan, "Reinforcement learning-based output feedback control of nonlinear systems with input constraints," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 35, no. 1, pp. 150–154, Feb. 2005.

[15] L. Yang, J. Si, K. S. Tsakalis, and A. A. Rodriguez, "Direct heuristic dynamic programming for nonlinear tracking control with filtered tracking error," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 6, pp. 1617–1622, Dec. 2009.

[16] D. P. Bertsekas, *Dynamic Programming and Optimal Control*. Belmont, MA: Athena Scientific, 2000.

[17] J. Si, A. G. Barto, W. B. Powell, and D. Wunsch, Eds., *Handbook of Learning and Approximate Dynamic Programming*. New York: Wiley-IEEE Press, 2004.

[18] B. Igelnik and Y. H. Pao, "Stochastic choice of basis functions in adaptive function approximation and the functional-link net," *IEEE Trans. Neural Networks*, vol. 6, no. 6, pp. 1320–1329, Nov. 1995.

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

[20] W. T. Miller, R. S. Sutton, and P. J. Werbos, Eds., *Neural Networks for Control*. Cambridge, MA: MIT Press, 1990.

[21] P. J. Werbos, "Building and understanding adaptive systems: A statistical/numerical approach to factory automation and brain research," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-17, no. 1, pp. 7–20, Jan. 1987.

[22] P. J. Werbos, "Advanced forecasting methods for global crisis warning and models of intelligence," *Gen. Syst. Yearbook*, vol. 22, pp. 25–38, 1977.

[23] T. Borgers and R. Sarin, "Learning through reinforcement and replicator dynamics," *J. Economic Theory*, vol. 77, no. 1, pp. 1–17, Nov. 1997.

[24] Q. Yang, J. B. Vance, and S. Jagannathan, "Control of nonaffine nonlinear discrete-time systems using reinforcement learning-based linearly parameterized neural networks," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 38, no. 4, pp. 994–1001, Aug. 2008.

[25] S. Jagannathan, "Discrete-time CMAC control of a feedback linearizable nonlinear systems under a persistence of excitation," *IEEE Trans. Neural Netw.*, vol. 10, no. 1, pp. 128–137, Jan. 1999.

[26] S. Jagannathan, *Neural Network Control of Nonlinear Discrete-time Systems*. Boca Raton, FL: CRC Press, 2006.

[27] 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.

[28] M. Krstic, I. Kanellakopoulos, and P. Kokotovic, *Nonlinear and Adaptive Control Design*. New York: Wiley, 1995.

[29] J. J. Murray, C. J. Cox, G. G. Lendaris, and R. Saeks, "Adaptive dynamic programming," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 32, no. 2, pp. 140–153, May 2002.

[30] K. J. Astrom and B. Wittenmark, *Adaptive Control.*, 2nd ed. Reading, MA: Addison-Wesley, 1994.

[31] 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.

Qinmin Yang received the B.S. degree in electrical engineering from Civil Aviation University of China, Tianjin, China, in 2001, the M.S. degree in control science and engineering from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2004, and the Ph.D. degree in electrical engineering from the University of Missouri, Rolla, in 2007.

From 2007 to 2008, he was a Postdoctoral Research Associate with the University of Missouri. In 2008, he was a System Engineer with Caterpillar Inc. From 2009 to 2010, he was a Postdoctoral Research Associate with the University of Connecticut, Storrs. Since 2010, he has been with the State Key Laboratory of Industrial Control Technology, Department of Control Science and Engineering, Zhejiang University, Hangzhou, China, where he is currently an Assistant Professor. His research interests include nanorobotics, intelligent control, neural networks, autonomous systems, and system diagnosis.

Sarangapani Jagannathan received the Ph.D. degree in electrical engineering from the University of Texas, Arlington, in 1994.

He was with the Systems and Controls Research Division, Caterpillar Inc., Peoria, IL, as a Consultant, from 1994 to 1998. From 1998 to 2001, he was with the University of Texas, San Antonio, as an Assistant Professor, and since September 2001, he has been with the Missouri University of Science and Technology (formerly University of Missouri, Rolla) where he is currently a Rutledge–Emerson Distinguished Professor and the Site Director for the National Science Foundation (NSF) Industry/University Cooperative Research Center on Intelligent Maintenance Systems. He has coauthored about 90 peer-reviewed journal articles, 180 refereed IEEE conference articles, several book chapters, and three books entitled "Neural Network Control of Robot Manipulators and Nonlinear Systems" (Taylor & Francis, 1999), "Discrete-Time Neural Network Control of Nonlinear Discrete-Time Systems" (CRC Press, 2006), and "Wireless Ad Hoc and Sensor Networks: Performance, Protocols and Control" (CRC Press, 2007). He is the holder of 18 patents with several pending. He, so far, supervised the completion of 14 doctoral students and 26 M.S. students. His research funding is in excess of 12 million dollars from NSF, National Aeronautics and Space Administration, Air Force Research Laboratory, Sandia, and from companies such as Boeing, Caterpillar, Chevron, Honeywell, and others. He is currently serving the U.K. *Transactions of Measurement and Control*. He is also serving as the Coeditor for the Institute of Engineering and Technology Book series on Control. His research interests include adaptive and neural network control, computer/communication/sensor networks, prognostics, and autonomous systems/robotics.

Dr. Jagannathan was the recipient of the NSF Career Award in 2000, Caterpillar Research Excellence Award in 2001, Boeing Pride Achievement Award in 2007, and many others. He served as the Associate Editor for IEEE TRANSACTIONS ON NEURAL NETWORKS, IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY, and IEEE SYSTEMS JOURNAL. He is also serving on a number of IEEE conference committees.