Correspondence

Direct Heuristic Dynamic Programming for Nonlinear Tracking Control With Filtered Tracking Error

Lei Yang, Jennie Si, Konstantinos S. Tsakalis, and Armando A. Rodriguez

***Abstract*—This paper makes use of the direct heuristic dynamic programming design in a nonlinear tracking control setting with filtered tracking error. A Lyapunov stability approach is used for the stability analysis of the tracking system. It is shown that the closed-loop tracking error and the approximating neural network weight estimates retain the property of uniformly ultimate boundedness under the presence of neural network approximation error and bounded unknown disturbances under certain conditions.**

***Index Terms*—Approximate dynamic programming (ADP), direct heuristic dynamic programming (direct HDP), Lyapunov stability, tracking control.**

I. Introduction

Dynamic programming (DP) is the approach to computing the optimal control policy over time under nonlinearity and uncertainty by employing the principle of optimality introduced by Bellman [1]. The common goal of approximate DP (ADP) [2] approaches is to address the question of how to use approximation methods and/or learning in a DP formulation to extend the power of the Bellman equation [1], so that widely applicable robust methods for obtaining nearly optimal control strategies can be developed. The direct heuristic DP (HDP) method studied herein is an ADP method that was introduced in [3], which was inspired by action-dependent HDP (ADHDP) [4]–[7], and has been applied to large-scale complex realistic applications [8], [9].

Developing ADP designs with desired performances, including convergence properties for large-scale complex realistic dynamic systems, has been a research focus of many researchers for many years. Most of the early results on ADP designs with convergence guarantees have been obtained by using discrete-time finite-state systems under a Markov decision process (MDP) setting. Either lookup tables such as Q-learning [11], [12] or linear function approximators such as linear TD(λ) [13] have been considered to establish convergence results. In [14], Konda and Tsitsiklis proposed a class of actor–critic algorithms with convergence guarantee, in which the critic uses temporal difference learning with a linearly parameterized approximation architecture for MDPs with polish (complete, separable, and metric) state and action spaces. Much effort has been made to extend the ADP designs to real-time continuous states and control systems with uncertainties, mainly by discretization and application of well-established discrete-time results. However, most of these ADP techniques do not scale up. Bradtke *et al.* first presented a proof to a linear system with linear quadratic regulation problems using a special form of Q-learning [15]. It was further extended using HDP and dual HDP (DHP) with a similar linear control structure by Landelius [16]. Anderson *et al.* proposed a method that combines a proportional–integral controller and reinforcement learning within a robust control framework. A Q-learning component was incorporated into the integral quadratic constraint framework, which was guaranteed to be stable to the predefined disturbance boundaries. The design has been implemented for heating, ventilation, and air conditioning control of buildings in [17]. A series of results [18]–[23] have demonstrated the applications of ADP designs to various advanced control problem settings, e.g., H_∞ control and zero-sum games. In [18], Al-Tamimi *et al.* used the HDP and DHP structures proposed in [4]–[7] to solve a discrete-time zero-sum game in which the state and action spaces were continuous. The design was tested on an H_∞ autopilot problem with convergence proof. The structure was extended later onto a Q-learning framework where no explicit model was required in [19]. In [20], Vrabie *et al.* proposed a continuous-time state feedback optimal controller based on an actor–critic structure for linear systems. Convergence properties were established by showing equivalence to the quasi-Newton search method. In [21], a neural network Hamilton–Jacobi–Bellman (HJB) approach was presented for nonlinear systems with saturating actuators by Abu-Khalaf *et al.* Neural networks were used to approximate the cost function associated with solving the HJB equation. ADP techniques such as policy iterations and zero-sum games were integrated into the design to solve H_∞ control of nonlinear systems with input saturation [22], [23].

This paper focuses on a generic tracking control structure using direct HDP as an online complementary learning control element. The aim is to address general nonlinear dynamic control problems with continuous states and uncertainty. Under the proposed controller structure, the tracking results are examined using a Lyapunov stability framework for a class of general nonlinear discrete-time systems. Nonlinear control systems with filtered tracking error have extensively been used in adaptive control literature [24]–[26]. Adaptive neural network controllers have been demonstrated in various problem settings as nonlinear controller designs in the field of feedback control [27]. Typically, the universal approximation property of neural networks is used to make a case for nonlinear control capability. In these approaches, classical adaptive control ideas are used, and tracking errors are formulated into short-term system performance measures. A similar control structure was adopted for an adaptive critic-based neural network control design for an engine emission control problem [28], [29]. In this approach, the neural-network-based controller design was similar to the original ASE-ACE structure in [10], with two similar building blocks called action and critic neural networks. The inputs to the critic contain the current states of the system, and the weights of the action network are updated based on the output of the critic network. For the design presented in this paper, however, the direct HDP implemented in the proposed tracking structure was inspired by the general ADHDP structure [6], [7] with several of its unique features [3], [30]. Both system states and control actions are included as inputs to the critic. The weights of the action network were adapted by gradient descent algorithms to reduce the squared error between the desired ultimate objective and the approximated overall cost, and this update is performed through the weights in the critic. By doing so, the tuning of the action network can directly be backpropagated via the

Manuscript received May 24, 2008; revised October 15, 2008 and January 15, 2009. First published May 29, 2009; current version published November 18, 2009. This work was supported in part by National Science Foundation under Grant ECS-0002098, Grant ECS-0233529, and Grant DUE-0231440. This paper was recommended by Associate Editor W. E. Dixon.

L. Yang is with the Department of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China (e-mail: y_lei@zju.edu.cn).

J. Si, K. S. Tsakalis, and A. A. Rodriguez are with Arizona State University, Tempe, AZ 85281 USA (e-mail: si@asu.edu; konstantinos.tsakalis@asu.edu; aar@asu.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/TSMCB.2009.2021950

control signal. In other words, the action network updates are guided by a gradient descent direction, and this is tied into the final control objective. Practically, we have observed robust performances in terms of learning controller numerical stability during training. Because of this, we were able to demonstrate successful applications of the direct HDP to large complex problems [8], [9].

In addition to proposing the generic online learning control structure, as will be presented next, the results obtained in this paper provide a comprehensive performance bound for the direct HDP learning system under a tracking problem setting for a general class of nonlinear dynamic systems.

II. System Description

Consider an nmth-order multiple-input–multiple-output discrete-time nonlinear system. Let $x(t) = [\,x_1(t) \quad x_2(t) \quad \cdots \quad x_n(t)\,]^T$ denote the system states. The nonlinear system dynamics are defined as

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

where $x(t) \in R^{nm}$ is the internal state vector, with $x_i(t) \in R^m$, and $u(t) \in R^m$ is the control input. Nonlinear function $\{f(\cdot) : R^{nm} \to R^m\}$ is assumed to be unknown. Variable $d(t) \in R^m$ is a disturbance acting on the system at time instance t, which is assumed unknown but bounded, so that $\|d(t)\| \le d_m$, where d_m is a known constant and $\|\cdot\|$ is the Euclidean vector 2-norm. This is the *discrete Brunovsky canonical form*. Many systems naturally occur in Brunovsky form [26]. Moreover, it is often possible to transform general discrete-time systems into discrete Brunovsky form [31].

III. Tracking Error Dynamics for a Class of Nonlinear Systems

Given a desired trajectory $x_d(t) \in R^m$ and its delayed values, tracking error $e(t)$ is defined as

$$e(t) = x_n(t) - x_d(t). \tag{2}$$

Filtered tracking error $\bar{e}(t) \in R^m$ is defined as

$$\bar{e}(t) = e(t) + \lambda_1 e_{n-1}(t) + \cdots + \lambda_{n-1} e_1(t) \tag{3}$$

where $e_{n-1}(t), \ldots, e_1(t)$ are the delayed errors of $e(t)$, i.e., $e_{n-k}(t) = e_n(t-k)$, and $\lambda_1, \ldots, \lambda_{n-1}$ are the constant matrices selected, so that $|z^{n-1} + \lambda_1 z^{n-2} + \cdots + \lambda_{n-1}|$ is stable. It can further be written into matrix form as

$$\bar{e}(t) = [\Lambda \quad I_{m\times m}] e(t) \tag{4}$$

where $e(t) = [e_1^T, e_2^T, \ldots, e_n^T]^T$, $\Lambda = [\lambda_{n-1} I_{m\times m}, \lambda_{n-2} I_{m\times m}, \ldots, \lambda_1 I_{m\times m}] \in R^{m\times(n-1)m}$, and $I_{m\times m} \in R^{m\times m}$ is an identity matrix.

Rewriting (3) at $t+1$, i.e.,

$$\bar{e}(t+1) = e(t+1) + \lambda_1 e_{n-1}(t+1) + \cdots + \lambda_{n-1} e_1(t+1)$$

and combining (3) with (1), the discrete-time nonlinear system can be written in terms of the filtered tracking error [26] as

$$\begin{aligned} \bar{e}(t+1) = f\left(x(t)\right) - x_d(t+1) + \lambda_1 e_n(t) + \cdots \\ + \lambda_{n-1} e_2(t) + u(t) + d(t). \end{aligned} \tag{5}$$

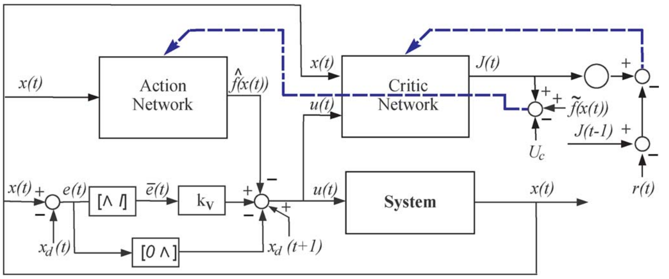

Fig. 1. Schematic diagram of direct HDP implementation for nonlinear tracking control with filtered error tracking.

IV. Basic Controller Design

Define control input $u(t)$ as

$$\begin{aligned} u(t) = x_d(t+1) - \hat{f}\left(x(t)\right) + k_v \bar{e}(t) \\ - \lambda_1 e_n(t) - \cdots - \lambda_{n-1} e_2(t) \end{aligned} \tag{6}$$

with a diagonal gain matrix k_v, and $\hat{f}(x(t))$ is an approximation of the unknown nonlinear function $f(x(t))$. The closed-loop error system then becomes

$$\bar{e}(t+1) = k_v \bar{e}(t) - \tilde{f}\left(x(t)\right) + d(t) \tag{7}$$

where $\tilde{f}(x(t))$ is the function approximation error given by

$$\tilde{f}\left(x(t)\right) = f\left(x(t)\right) - \hat{f}\left(x(t)\right).$$

Assuming that function approximation error $\tilde{f}(x(t))$ and unknown disturbance $d(t)$ are bounded, the filtered tracking error system is stable, given $0 < k_{\mathrm{vmax}} < 1$, where k_{vmax} is the maximum eigenvalue of the constant gain matrix k_v [26].

V. Direct HDP for Nonlinear Tracking Control With Filtered Tracking Error

A general schematic diagram of applying the direct HDP to the nonlinear tracking control problem with filtered tracking error is shown in Fig. 1.

A. *Reinforcement Signal*

Binary reinforcement signal $r(t)$ is defined based on the current filtered tracking error $\bar{e}(t)$ as

$$r(t) = [r_1(t), r_2(t), \ldots, r_m(t)] \in R^m \tag{8}$$

with

$$r_i(t) = \begin{cases} 0, & \text{if } \|\bar{e}_i(t)\| \le c \\ 1, & \text{if } \|\bar{e}_i(t)\| > c \end{cases}, \quad i = 1, 2, \ldots, m \tag{9}$$

where constant c is a predefined threshold for the filtered tracking error, and $\|\cdot\|$ is the Euclidean vector 2-norm. Signal $r_i(t)$ is provided from the external environment and is as simple as either a "0" or a "1," corresponding to "good" or "poor" tracking performance, respectively.

The goal of direct HDP control is to optimize a weighted total cost-to-go objective $J(t)$ given by

$$J(t) = r(t+1) + \alpha r(t+2) + \alpha^2 r(t+3) \cdots \tag{10}$$

where $\alpha > 0$ is a constant weighting factor.

B. Critic Network

By writing the Bellman equation into the prediction error of the critic element

$$e_c(t) = \alpha J(t) - [J(t-1) - r(t)] \tag{11}$$

the square of this error provides the desired objective function for the critic network to minimize by tuning critic network weights.

The critic network is implemented as a feedforward neural network with one hidden layer, which is defined as follows:

$$J(t) = \sum_{i=1}^{N_h} \hat{w}_{c_i}^{(2)}(t) p_i(t) \tag{12}$$

with

$$\begin{aligned} p_i(t) &= \frac{1-\exp^{-q_i(t)}}{1+\exp^{-q_i(t)}} \\ q_i(t) &= \sum_{j=1}^{n+1} \hat{w}_{c_{ij}}^{(1)}(t) x_j(t) \\ &= \sum_{j=1}^{n} \hat{w}_{c_{ij}}^{(1)}(t) x_j(t) + \hat{w}_{c_{in+1}}^{(1)}(t) u(t), \\ &\qquad i = 1, \ldots, N_h \end{aligned} \tag{13}$$

where $\hat{w}_{c_i}^{(2)}(t)$ and $\hat{w}_{c_i}^{(1)}(t)$ are the weights for the output and hidden layers, respectively; q_i is the ith hidden node input of the critic network; and p_i is the corresponding output of the hidden node. N_h is the total number of hidden nodes in the critic network, and $n+1$ is the total number of inputs into the critic network, including the analog action value $u(t)$ from the action network.

Writing (12) into matrix form, we have

$$J(t) = \hat{w}_c^T(t) \phi_c\left(w_{\text{ch}}^T x(t)\right) = \hat{w}_c^T(t) \phi_c(t) \tag{14}$$

where $\hat{w}_c(t)$ and w_{ch} are the weight matrices of the output and hidden layers of the critic network, respectively. Since only the output weights $\hat{w}_c(t)$ are updated during the iterations, with the hidden weights w_{ch} fixed, function $\phi_c(w_{\text{ch}}^T x(t))$ is written in $\phi_c(t)$ as the vector activation function in the hidden layer.

The objective function to be minimized in the critic network is

$$E_c(t) = \frac{1}{2} e_c^T(t) e_c(t). \tag{15}$$

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

$$\begin{aligned} \hat{w}_c(t+1) &= \hat{w}_c(t) + \Delta \hat{w}_c(t) \\ \Delta \hat{w}_c(t) &= l_c \left[-\frac{\partial E_c(t)}{\partial \hat{w}_c(t)} \right] \\ &= l_c \left[-\frac{\partial E_c(t)}{\partial J(t)} \frac{\partial J(t)}{\partial \hat{w}_c(t)} \right] \end{aligned}$$

where $l_c > 0$ is the learning rate of the critic network.

By substituting (11) into the preceding equation, the weight-updating rule becomes

$$\begin{aligned} \hat{w}_c(t+1) = \hat{w}_c(t) - l_c \phi_c(t) \big(& \alpha \hat{w}_c^T(t) \phi_c(t) + r(t) \\ & - \hat{w}_c^T(t-1) \phi_c(t-1) \big)^T. \end{aligned} \tag{16}$$

C. Action Network

In direct HDP [3], the action network aims at learning an optimal controller under the constraint of the principle optimality. In the current tracking application using direct HDP, due to the problem construction described by (6), the action network output is to approximate the dynamic system denoted by $f(\cdot)$. Letting function approximation error $\tilde{f}(t)$ be

$$\tilde{f}(t) = \hat{f}(t) - f(t) \tag{17}$$

where $\hat{f}(t)$ is the action network output, and $f(t)$ is the unknown nonlinear system dynamics, the prediction error for the action element is defined as

$$e_a(t) = J(t) - U_c(t) + \tilde{f}(t). \tag{18}$$

Note that, if function approximation error $\tilde{f}(t)$ approaches 0, i.e., a perfect model of the nonlinear system is obtained by the action NN, then, according to the closed-loop error system in (7)

$$\tilde{f}\left(x(t)\right) = k_v \bar{e}(t) - \bar{e}(t+1) + d(t) \tag{19}$$

the filtered tracking error will also approach 0, i.e., the optimal performance with respect to the cost will also be achieved. In fact, proving that both parts of error term $e_a(t)$ are uniformly ultimate bounded (UUB) will be the main result of this paper.

In the current paradigm and without loss of generality, the term U_c is set to "0," corresponding to "success." Thus, the action network error function becomes

$$e_a(t) = J(t) + \tilde{f}(t). \tag{20}$$

Similar to the critic network, the action network is implemented using a feedforward neural network with one hidden layer. Its matrix form can be expressed as

$$\hat{f}(t) = \hat{w}_a^T(t) \phi_a\left(w_{ah}^T x(t)\right) = \hat{w}_a^T(t) \phi_a(t) \tag{21}$$

where $\hat{w}_a(t)$ and w_{ah} are the weight matrices of the output and hidden layers, respectively. Similar to the critic network, weights w_{ah} are fixed during the adaptation, and $\phi_a(t)$ is used as the vector activation function in the hidden layer.

Assuming that the unknown system dynamics can accurately be represented by a neural network, i.e., suppose

$$f(t) = w_a^T(t) \phi_a(t) + \varepsilon_a\left(x(t)\right) \tag{22}$$

where $\varepsilon_a(x(t))$ is the neural network function reconstruction error vector, function approximation error $\tilde{f}(t)$ can then be written as

$$\tilde{f}(t) = \left(\hat{w}_a(t) - w_a(t)\right)^T \phi_a(t) - \varepsilon_a\left(x(t)\right). \tag{23}$$

The objective function to be minimized in the action network is

$$E_a(t) = \frac{1}{2} e_a^T(t) e_a(t). \tag{24}$$

The weights in the action network are updated to minimize the square of this performance error. For finding the tuning law, the formulation of $\tilde{f}(t)$ given by (7) is considered. The update rule is then similar to that in the critic network by gradient descent and/or its variants

$$\begin{aligned} \hat{w}_a(t+1) &= \hat{w}_a(t) + \Delta \hat{w}_a(t) \\ \Delta \hat{w}_a(t) &= l_a \left[-\frac{\partial E_a(t)}{\partial \hat{w}_a(t)} \right] \\ &= l_a \left[-\frac{\partial E_a(t)}{\partial J(t)} \frac{\partial J(t)}{\partial u(t)} \frac{\partial u(t)}{\partial \hat{w}_a(t)} \right] \end{aligned}$$

where $l_a > 0$ is the learning rate of the action network at time instance k.

By substituting (18), we have

$$\hat{w}_a(t+1) = \hat{w}_a(t) - l_a \omega_{c,n+1} \phi_a(t) \left(\hat{w}_c^T(t)\phi_c(t) + \tilde{f}(t)\right)^T$$

where $\omega_{c,n+1}$ is the absolute value of the critic weight connected to control input u from the action network. Considering (19), the updating rule becomes

$$\hat{w}_a(t+1) = \hat{w}_a(t) - l_a \omega_{c,n+1} \phi_a(t) \times \left(\hat{w}_c^T(t)\phi_c(t) + k_v \bar{e}(t) - \bar{e}(t+1)\right)^T \quad (25)$$

with bounded disturbance $d(t)$ taken to zero.

VI. Lyapunov Stability Analysis

Prior to the overall system stability analysis, the following two assumptions are made, which can reasonably be satisfied under the current problem settings. These assumptions are usually given in neural network control designs to bypass the technically involved aspects of the analysis and concentrate on the structural properties of the algorithm [26], [28], [29], which is our main interest here. In the current formulation, $\|\cdot\|_1$ represents the vector 1-norm, and $\|\cdot\|$ stands for the Euclidean vector 2-norm if not specified elsewhere.

Assumption 6.1: Bounded optimal network weights: With w_c and w_a as the optimal weights for the critic and action networks, respectively, assume that they are bounded, so that

$$\|w_a\|_1 \le w_{cm}, \quad \|w_a\|_1 \le w_{am} \quad (26)$$

where w_{cm}, $w_{am} \in R$ are constant bounds on the unknown network weights.

Assumption 6.2: Bounded function reconstruction error vector: Assume that the function reconstruction error vector is bounded, so that

$$\|\varepsilon_a(x(t))\|_1 \le \varepsilon_{am} \quad (27)$$

over some compact set S, where $x(t) \in S$, and $\varepsilon_{am} \in R$ is a constant bound on the function reconstruction error vector.

In the rest of this section, some properties of the filtered tracking error, i.e., the weights of the critic and action networks, will be first given in Lemma (6.1)–(6.3) under the current problem settings. Then, the final proof of the controller stability will be presented by Theorem (6.4).

Lemma 6.1: Let the desired trajectory $x_d(t)$ be uniformly bounded and Assumptions 6.1 and 6.2 hold. Take the control input as (6) and the action network setting as (21); then, for

$$J_1(t) = \frac{1}{\gamma_1} \bar{e}(t)^T \bar{e}(t) \quad (28)$$

its first difference is given by

$$\Delta J_1(t) \le \frac{3}{\gamma_1}\left[\left(k_{\text{vmax}}^2 - \frac{1}{3}\right)\|\bar{e}(t)\|^2 + \|\zeta_a(t)\|^2 + \|\varepsilon_a(t) + d(t)\|^2\right] \quad (29)$$

where $\zeta_a(t) = (\hat{w}_a(t) - w_a(t))^T \phi_a(t) = \tilde{w}_a^T(t)\phi_a(t)$ is the approximation error of the action network, and $\gamma_1 > 0$ is a weighting factor.

Proof: By substituting the tracking error dynamics

$$\bar{e}(t+1) = k_v \bar{e}(t) + \tilde{f}(x(t)) + d(t)$$

the first difference becomes

$$\Delta J_1(t) = \frac{1}{\gamma_1}\Big[(k_v \bar{e}(t) - \zeta_a(t) + \varepsilon_a(t) + d(t))^T \times (k_v \bar{e}(t) - \zeta_a(t) + \varepsilon_a(t) + d(t)) - \bar{e}(t)^T \bar{e}(t)\Big]. \quad (30)$$

Applying Cauchy–Schwarz inequality, we have (29).

Lemma 6.2: Let the desired trajectory $x_d(t)$ be uniformly bounded and Assumptions 6.1 and 6.2 hold. Take the control input as (6), the reinforcement signal as (9), the critic network setting as (14), and its network updating rules as (16); then, for

$$J_2(t) = \frac{1}{l_c} tr\left(\tilde{w}_c^T(t)\tilde{w}_c(t)\right) \quad (31)$$

its first difference is given by

$$\begin{aligned}\Delta J_2(t) \le & -\alpha\|\zeta_c(t)\|^2 - \alpha\left(I - l_c \alpha \phi_c(t)\phi_c^T(t)\right) \\ & \cdot \Big\|\zeta_c(t) + w_c^T(t)\phi_c(t) + \alpha^{-1} r(t) \\ & \quad - \alpha^{-1}\hat{w}_c^T(t-1)\phi_c(t-1)\Big\|^2 \\ & + 2\alpha^{-1}\left\|\alpha w_c^T(t)\phi_c(t) + r(t) - w_c^T\phi_c(t-1)\right\|^2 \\ & + 2\alpha^{-1}\|\zeta_c(t-1)\|^2 \end{aligned} \quad (32)$$

where $\zeta_c(t) = (\hat{w}_c(t) - w_c(t))^T \phi_c(t) = \tilde{w}_c^T(t)\phi_c(t)$ is the approximation error of the critic network.

Proof: The first difference can be written as

$$\Delta J_2(t) = \frac{1}{l_c} tr\left(\tilde{w}_c^T(t+1)\tilde{w}_c(t+1) - \tilde{w}_c^T(t)\tilde{w}_c(t)\right). \quad (33)$$

With the weight updating rule of $w_c(t+1)$ as in (16), we get

$$\tilde{w}_c(t+1) = \left(I - l_c \alpha \phi_c(t)\phi_c^T(t)\right)\tilde{w}_c(t) - l_c \phi_c(t) \times \left(\alpha w_c^T \phi_c(t) + r(t) - \hat{w}_c^T(t-1)\phi_c(t-1)\right)^T.$$

Substituting it into (33), we get

$$\Delta J_2(t) = \frac{1}{l_c} tr(P_1 + P_2 + P_3). \quad (34)$$

For P_1, we have

$$P_1 = -l_c \alpha \|\zeta_c(t)\|^2 - l_c \alpha \tilde{w}_c^T(t) \times \left(I - l_c \alpha \phi_c(t)\phi_c^T(t)\right)\phi_c(t)\phi_c^T(t)\tilde{w}_c(t).$$

For P_2

$$P_2 = -2 l_c \tilde{w}_c^T(t)\left(I - l_c \alpha \phi_c(t)\phi_c^T(t)\right)\phi_c(t) \times \left(\alpha w_c^T \phi_c(t) + r(t) - \hat{w}_c^T(t-1)\phi_c(t-1)\right)^T.$$

Then, for P_3

$$P_3 = l_c^2 \alpha^2 \phi_c^T(t)\phi_c(t)\Big\|w_c^T(t)\phi_c(t) + \alpha^{-1} r(t) - \alpha^{-1}\hat{w}_c^T(t-1)\phi_c(t-1)\Big\|^2.$$

Combining the three terms and applying the Cauchy–Schwarz inequality with

$$\hat{w}_c^T(t-1)\phi_c(t-1) = w_c^T \phi_c(t-1) + \tilde{w}_c^T(t-1)\phi_c(t-1)$$

we have (32).

Lemma 6.3: Let the desired trajectory $x_d(t)$ be uniformly bounded and Assumptions 6.1 and 6.2 hold. Take the control input as (6), the reinforcement signal as (9), the critic network setting as (14), the action network setting as (21), and the network updating rules as (16) and (25); then, for

$$J_4(t) = \frac{1}{\gamma_3 l_a} tr\left(\tilde{w}_a^T(t)\tilde{w}_a(t)\right) \qquad (35)$$

its first difference is given by

$$\begin{aligned}\Delta J_4(t) \leq & -\frac{1}{\gamma_3}\left\{ \left(\omega_{c,n+1} - l_a w_{c,n+1}^2 \phi_a^T(t)\phi_a(t)\right) \right.\\ & \left. \times \left\|\hat{w}_c^T(t)\phi_c(t) + \zeta_a(t) - \varepsilon_a(t) - d(t)\right\|^2 \right\}\\ & + \frac{2}{\gamma_3}\omega_{c,n+1}\left\|w_c^T(t)\phi_c(t) - (\varepsilon_a(t) + d(t))\right\|^2\\ & - \frac{1}{\gamma_3}\omega_{c,n+1}\left\|\zeta_a(t)\right\|^2 + \frac{2}{\gamma_3}\omega_{c,n+1}\left\|\zeta_c(t)\right\|^2 \end{aligned} \qquad (36)$$

where $\zeta_a(t)$ and $\zeta_c(t)$ are the approximation errors of the action and critic networks, respectively, and $\gamma_3 > 0$ is a weighting factor.

Proof: The first difference can be written as

$$\Delta J_4(t) = J_4(t+1) - J_4(t) \qquad (37)$$

with the weight updating rule of $w_a(t+1)$ as in (25) and the tracking error dynamics as

$$k_v\bar{e}(t) - \bar{e}(t+1) = \zeta_a(t) - \varepsilon_a(t) - d(t).$$

We have

$$\begin{aligned}\tilde{w}_a(t+1) = & \tilde{w}_a(t) - l_a\omega_{c,n+1}\phi_a(t)\\ & \times\left(\hat{w}_c^T(t)\phi_c(t) + \zeta_a(t) - \varepsilon_a(t) - d(t)\right)^T. \end{aligned} \qquad (38)$$

Substituting the preceding equation into (37), with $\zeta_a(t) = \tilde{w}_a^T(t)\phi_a(t)$, it can be written as

$$\begin{aligned}&\Delta J_4(t)\\ &= \frac{1}{\gamma_3}\left\{ -\left(\omega_{c,n+1} - l_a w_{c,n+1}^2\phi_a^T(t)\phi_a(t)\right)\right.\\ &\qquad \times\left\|\hat{w}_c^T(t)\phi_c(t) + \zeta_a(t) - (\varepsilon_a(t) + d(t))\right\|^2\\ &\qquad -\omega_{c,n+1}\left\|\zeta_a(t)\right\|^2 + \omega_{c,n+1}\\ &\qquad \left.\times\left\|w_c^T(t)\phi_c(t) - (\varepsilon_a(t) + d(t)) + \tilde{w}_c^T(t)\phi_c(t)\right\|^2\right\}. \end{aligned} \qquad (39)$$

Applying Cauchy–Schwarz inequality, together with $\zeta_c(t) = \tilde{w}_c^T(t)\phi_c(t)$, we have (36). ■

Theorem 6.4: Let the desired trajectory $x_d(t)$ be uniformly bounded and Assumptions 6.1 and 6.2 hold. Take the control input as (6), the reinforcement signal as (9), the critic network setting as (14), the action network setting as (21), and the network updating rules as (16) and (25); then, filtered tracking error $\bar{e}(t)$ and network weight estimates $\tilde{w}_c(t)$ and $\tilde{w}_a(t)$ are UUB, provided the following condition holds:

$$\begin{aligned}& l_c\left\|\phi_c(t)\right\|^2 < 1 \quad l_a\left\|\phi_a(t)\right\|^2 < \frac{1}{\omega_{c,n+1}}\\ & \alpha > \sqrt{2} \quad 0 < k_{\text{vmax}} < \frac{\sqrt{3}}{3}\end{aligned}$$

where k_{vmax} is the maximum eigenvalue of the constant gain matrix k_v, and $\omega_{c,n+1}$ is the absolute value of the critic weight $w_{c,n+1}$ connected to the control input u from the action network.

Proof: Define the Lyapunov function candidate as

$$\begin{aligned}J(t) = & \frac{1}{\gamma_1}\bar{e}(t)^T\bar{e}(t) + \frac{1}{l_c}tr\left(\tilde{w}_c^T(t)\tilde{w}_c(t)\right)\\ & + \frac{1}{\gamma_2}\left\|\zeta_c(t-1)\right\|^2 + \frac{1}{\gamma_3 l_a}tr\left(\tilde{w}_a^T(t)\tilde{w}_a(t)\right)\end{aligned} \qquad (40)$$

where $\gamma_1, \gamma_2, \gamma_3 > 0$.

Let the first difference be

$$\Delta J(t) = \Delta J_1(t) + \Delta J_2(t) + \Delta J_3(t) + \Delta J_4(t). \qquad (41)$$

Write $\Delta J_3(t)$ as

$$\Delta J_3(t) = \frac{1}{\gamma_2}\left(\left\|\zeta_c(t)\right\|^2 - \left\|\zeta_c(t-1)\right\|^2\right) \qquad (42)$$

together with Lemma (6.1)–(6.3). By assumption, select

$$\begin{aligned}& l_c\left\|\phi_c(t)\right\|^2 < 1 \quad l_a\left\|\phi_a(t)\right\|^2 < \frac{1}{\omega_{c,n+1}}\\ & \alpha > \sqrt{2} \quad 0 < k_{\text{vmax}} < \frac{\sqrt{3}}{3}\end{aligned} \qquad (43)$$

and select $\gamma_{\{1,2,3\}}$ satisfying

$$\begin{aligned}& \gamma_1 > 3\gamma_3/\omega_{c,n+1} \quad \gamma_2 = \alpha/2\\ & \gamma_3 > 2\alpha\omega_{c,n+1}/(\alpha^2 - 2)\end{aligned} \qquad (44)$$

such that

$$\begin{aligned}\Delta J(t) \leq & -\frac{1}{\gamma_1}\left(1 - 3k_{\text{vmax}}^2\right)\left\|\bar{e}(t)\right\|^2\\ & - \left(\alpha - \frac{1}{\gamma_2} - \frac{2}{\gamma_3}\omega_{c,n+1}\right)\left\|\zeta_c(t)\right\|^2\\ & - \left(\frac{1}{\gamma_3}\omega_{c,n+1} - \frac{3}{\gamma_1}\right)\left\|\zeta_a(t)\right\|^2\\ & - \alpha\left(I - l_c\alpha\phi_c(t)\phi_c^T(t)\right)\\ & \times\left\|\zeta_c(t) + w_c^T(t)\phi_c(t) + \alpha^{-1}r(t)\right.\\ & \qquad \left. -\alpha^{-1}\hat{w}_c^T(t-1)\phi_c(t-1)\right\|^2\\ & - \frac{1}{\gamma_3}\left(\omega_{c,n+1} - l_a w_{c,n+1}^2\phi_a^T(t)\phi_a(t)\right)\\ & \times\left\|\zeta_a(t) + \hat{w}_c^T(t)\phi_c(t) - (\varepsilon_a(t) + d(t))\right\|^2 + D^2\end{aligned}$$

where D^2 is defined as

$$\begin{aligned}D^2 = & 2\alpha^{-1}\left\|\alpha w_c^T(t)\phi_c(t) + r(t) - w_c^T\phi_c(t-1)\right\|^2\\ & + \frac{2}{\gamma_3}\omega_{c,n+1}\left\|w_c^T(t)\phi_c(t) - (\varepsilon_a(t) + d(t))\right\|^2\\ & + \frac{3}{\gamma_1}\left\|\varepsilon_a(t) + d(t)\right\|^2.\end{aligned} \qquad (45)$$

Applying Cauchy–Schwarz inequality, we have

$$\begin{aligned} D^2 \leq\, & 6\left(\alpha+\alpha^{-1}+\frac{1}{\gamma_3}\omega_{c,n+1}\right) w_{cm}^2\phi_{cm}^2 \\ & + 6\left(\frac{1}{\gamma_1}+\frac{1}{\gamma_3}\omega_{c,n+1}\right)\left(\varepsilon_{am}^2+d_m^2\right)+6\alpha^{-1} \\ =\, & D_m^2 \end{aligned}$$

where w_{cm}, ϕ_{cm}, ε_{am}, and d_m are the upper bounds of $\|w_c\|$, $\|\phi_c\|$, $\|\varepsilon_a\|$, and $\|d\|$, respectively.

Given that conditions (43) hold and

$$\begin{aligned} \|\bar{e}(t)\| &> \sqrt{\frac{\gamma_1}{1-3k_{\mathrm{vmax}}^2}D_m^2} \\ \|\zeta_c(t)\| &> \sqrt{\frac{1}{\alpha-\frac{1}{\gamma_2}-\frac{2}{\gamma_3}\omega_{c,n+1}}D_m^2} \\ \|\zeta_a(t)\| &> \sqrt{\frac{1}{\frac{1}{\gamma_3}\omega_{c,n+1}-\frac{3}{\gamma_1}}D_m^2} \end{aligned} \tag{46}$$

the first difference $\Delta J(t) \leq 0$. According to the standard Lyapunov extension theorem [32], this demonstrates that filtered tracking error $\bar{e}(t)$ and network weight estimates $\tilde{w}_c(t)$ and $\tilde{w}_a(t)$ are UUB. ■

VII. Conclusion

ADP algorithms have extensively been studied regarding their structure, measured learning efficiency, value function approximation bounds, and convergence rates, using learning statistics, due to the statistical learning nature of such approaches. However, less has been done in evaluating their convergence properties under general nonlinear system settings. Existing studies are mostly limited to discrete-time finite-state systems that either are with lookup tables or consider linear systems. This paper has studied the stability properties of our previously proposed direct HDP design for a set of general nonlinear discrete-time continuous systems. The control system framework of the nonlinear tracking control with filtered tracking error has been implemented for direct HDP design and analyzed using the Lyapunov stability approach. Results have shown that the closed-loop tracking error and the weight estimates of both critic and action networks in the direct HDP design are UUB under the presence of neural network approximation errors and bounded unknown disturbances under certain conditions.

References

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

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

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

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

[5] P. J. Werbos, "A menu of design for reinforcement learning over time," in *Neural Networks for Control*, W. T. Miller, III, R. S. Sutton, and P. J. Werbos, Eds. Cambridge, MA: MIT Press, 1990, ch. 3, pp. 67–95.

[6] P. J. Werbos, "Neurocontrol and supervised learning: An overview and valuation," in *Handbook of Intelligent Control*, D. White and D. Sofge, Eds. New York: Van Nostrand, 1992, pp. 65–89.

[7] P. J. Werbos, "Approximate dynamic programming for real-time control and neural modeling," in *Handbook of Intelligent Control*, D. White and D. Sofge, Eds. New York: Van Nostrand, 1992, pp. 493–525.

[8] R. Enns and J. Si, "Helicopter trimming and tracking control using direct neural dynamic programming," *IEEE Trans. Neural Netw.*, vol. 14, no. 4, pp. 929–939, Jul. 2003.

[9] C. Lu, J. Si, and X. Xie, "Direct heuristic dynamic programming for damping oscillations in a large power system," *IEEE Trans. Syst., Man, Cybern. B*, vol. 38, no. 4, pp. 1008–1013, Aug. 2008.

[10] 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. 834–846, Sep./Oct. 1983.

[11] C. J. C. H. Watkins, "Learning from delayed rewards," Ph.D. dissertation, Univ. Cambridge, Cambridge, U.K., 1989.

[12] J. N. Tsitsiklis, "Asynchronous stochastic approximation and Q-learning," *Mach. Learn.*, vol. 16, no. 3, pp. 185–202, Sep. 1994.

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

[14] V. R. Konda and J. N. Tsitsiklis, "Actor–critic algorithms," *SIAM J. Control Optim.*, vol. 42, no. 4, pp. 1143–1166, 2003.

[15] S. J. Bradtke, B. E. Ydstie, and A. G. Barto, "Adaptive linear quadratic control using policy iteration," in *Proc. Amer. Control Conf.*, 1994, pp. 3475–3479.

[16] T. Landelius, "Reinforcement learning and distributed local model synthesis," Ph.D. dissertation, Linköping Univ., Linko¨ping, Sweden, 1997.

[17] C. W. Anderson, D. Hittle, M. Kretchmar, and P. Young, "Robust reinforcement learning for heating, ventilation, and air conditioning control of buildings," in *Handbook of Learning and Approximate Dynamic Programming*, J. Si, A. G. Barto, W. B. Powell, and D. C. Wunsch, Eds. New York: Wiley, 2004, ch. 20.

[18] A. Al-Tamimi, M. Abu-Khalaf, and F. L. Lewis, "Adaptive critic designs for discrete-time zero-sum games with application to H_∞ control," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 37, no. 1, pp. 240–247, Feb. 2007.

[19] A. Al-Tamimi, F. L. Lewis, and M. Abu-Khalaf, "Model-free *Q*-learning designs for linear discrete-time zero-sum games with application to H-infinity control," *Automatica*, vol. 43, no. 3, pp. 473–481, Mar. 2007.

[20] D. Vrabie, F. L. Lewis, and M. Abu-Khalaf, "Biologically inspired scheme for continuous-time approximate dynamic programming," *Trans. Inst. Meas. Control*, vol. 30, no. 3/4, pp. 207–223, Aug. 2008.

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

[22] M. Abu-Khalaf, F. L. Lewis, and J. Huang, "Policy iterations and the Hamilton–Jacobi–Isaacs equation for H_∞ state feedback control with input saturation," *IEEE Trans. Autom. Control*, vol. 51, no. 12, pp. 1989–1995, Dec. 2006.

[23] M. Abu-Khalaf and F. L. Lewis, "Neurodynamic programming and zero-sum games for constrained control systems," *IEEE Trans. Neural Netw.*, vol. 19, no. 7, pp. 1243–1252, Jul. 2008.

[24] K. S. Narendra and K. S. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Trans. Neural Netw.*, vol. 1, no. 1, pp. 4–27, Mar. 1990.

[25] A. J. Calise, "Neural networks in nonlinear aircraft flight control," *IEEE Aerosp. Electron. Syst. Mag.*, vol. 11, no. 7, pp. 5–10, Jul. 1996.

[26] F. W. Lewis, S. Jagannathan, and A. Yesildirak, *Neural Networks Control of Robot Manipulators and Nonlinear Systems*. London, U.K.: Taylor & Francis, 1998.

[27] F. L. Lewis, J. P. Huang, D. V. T. Prokhorov, and D. C. Wunsch, "Guest editorial special issue on neural nets for feedback control systems," *IEEE Trans. Neural Netw.*, vol. 18, no. 4, pp. 969–972, Jul. 2007.

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

[29] J. B. Vance, A. Singh, B. C. Kaul, S. Jagannathan, and J. A. Drallmeier, "Neural network controller development and implementation for spark ignition engines with high EGR levels," *IEEE Trans. Neural Netw.*, vol. 18, no. 4, pp. 1083–1100, Jul. 2007.

[30] L. Yang, J. Si, K. Tsakalis, and A. Rodriguez, "Understand direct NDP with linear quadratic regulation," in *Proc. IEEE Int. Symp. Intell. Control*, 2004, pp. 374–379.

[31] J. C. Kalkkuhl and K. J. Hunt, "Discrete-time neural model structures for continuous-time nonlinear systems," in *Neural Adaptive Control Technology*, R. Zbikowsky and K. J. Hunt, Eds. Singapore: World Scientific, 1996.

[32] K. J. Astrom and B. Wittenmark, *Adaptive Control*. Reading, MA: Addison-Wesley, 1989.