Neural Networks 22 (2009) 851–860

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

2009 Special Issue

Optimal control of unknown affine nonlinear discrete-time systems using offline-trained neural networks with proof of convergence☆

Travis Dierks*, Balaje T. Thumati, S. Jagannathan

Department of Electrical and Computer Engineering, Missouri University of Science and Technology (formerly University of Missouri-Rolla), 1870 Miner Circle, Rolla, MO 65409, United States

ARTICLE INFO

Article history:
Received 6 May 2009
Received in revised form 8 June 2009
Accepted 25 June 2009

Keywords:
Nonlinear optimal control
Heuristic dynamic programming
System identification
Neural network

ABSTRACT

The optimal control of linear systems accompanied by quadratic cost functions can be achieved by solving the well-known Riccati equation. However, the optimal control of nonlinear discrete-time systems is a much more challenging task that often requires solving the nonlinear Hamilton–Jacobi–Bellman (HJB) equation. In the recent literature, discrete-time approximate dynamic programming (ADP) techniques have been widely used to determine the optimal or near optimal control policies for affine nonlinear discrete-time systems. However, an inherent assumption of ADP requires the value of the controlled system one step ahead and at least partial knowledge of the system dynamics to be known. In this work, the need of the partial knowledge of the nonlinear system dynamics is relaxed in the development of a novel approach to ADP using a two part process: online system identification and offline optimal control training. First, in the system identification process, a neural network (NN) is tuned online using novel tuning laws to learn the complete plant dynamics so that a local asymptotic stability of the identification error can be shown. Then, using only the learned NN system model, offline ADP is attempted resulting in a novel optimal control law. The proposed scheme does not require explicit knowledge of the system dynamics as only the learned NN model is needed. The proof of convergence is demonstrated. Simulation results verify theoretical conjecture.

1. Introduction

Unlike the optimal control of linear systems, the optimal control of nonlinear discrete-time systems often requires solving the nonlinear Hamilton–Jacobi–Bellman (HJB) equation. In practice, the discrete-time HJB equation is more difficult to work with than the Riccati equation because it involves solving nonlinear partial difference equations; therefore, several works in the literature have attempted to solve the discrete-time nonlinear optimal control problem using dynamic programming-based approaches and neural networks (NNs) (Al-Tamimi, Lewis, & Abu-Khalaf, 2008; Chen & Jagannathan, 2008; Prokhorov & Wunsch, 1997; Si, Barto, Powell & Wunsch, 2004) by assuming that there are no NN reconstruction errors. Several neural network adaptive critic designs are presented by Prokhorov and Wunsch (1997) where the NN approximation errors are not considered and the neural network weights are tuned using the back propagation algorithm. However, it has been shown in the literature (Jagannathan, 2006) that the back propagation algorithm does not always converge or remain bounded. Additionally, the proof that each design convergences to the optimal control is not presented.

An iterative solution to the generalized HJB equation is proposed by Chen and Jagannathan (2008) and a nearly optimal state feedback control law for affine nonlinear discrete-time systems is derived. A single NN is utilized to learn the cost function while the NN reconstruction errors are considered negligible. Recently, heuristic dynamic programming (HDP) (Si et al., 2004) techniques were utilized by Al-Tamimi et al. (2008) to develop an iteration-based solution to the HJB, and it was shown that the algorithm converges to the optimal control policy and the optimal value function that solves the HJB equation. Two NNs are utilized in the implementation of this algorithm: one to learn the optimal control policy and one to learn the optimal cost function. Again, the NN reconstruction errors of the NNs are ignored. Additionally, the internal dynamics are considered unknown whereas the control coefficient matrix has to be known.

In many of the previous dynamic programming works (Al-Tamimi et al., 2008; Chen & Jagannathan, 2008), affine nonlinear discrete-time systems of the form $x(k+1) = f(x(k)) + g(x(k))u(x(k))$ are considered, and the optimal control policies

☆ Research supported in part by NSF ECCS#0621924, GAANN Program and Intelligent Systems Center.

* Corresponding author. Tel.: +1 618 317 1354.
E-mail address: tad5x4@mst.edu (T. Dierks).

0893-6080/$ – see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.neunet.2009.06.014

weight matrices, $\varphi(\bar{z}(k))$ is the NN activation function, $\bar{z}(k) = v_s^{\mathrm{T}} z(k)$, $z(k) = [x(k)^{\mathrm{T}}\ u(k)^{\mathrm{T}}]^{\mathrm{T}}$ is the NN input, and $\varepsilon_s(k)$ is the bounded NN functional approximation error. In this work, the NN activation functions are selected to be hyperbolic tangent functions. Additionally, bounds on the output layer weights are taken as $\|w_s\|_F \leq W_{sm}$ for a constant W_{sm} while the NN activation functions are bounded such that $\|\varphi(z(k))\| \leq \varphi_M$ for a constant φ_M (Jagannathan, 2006). Using the NN representation, the system dynamics (4) become

$$x(k+1) = w_s^{\mathrm{T}}\varphi(v_s^{\mathrm{T}} z(k)) + \varepsilon_s(k) = w_s^{\mathrm{T}}\varphi(\bar{z}(k)) + \varepsilon_s(k). \tag{5}$$

The NN system identification scheme is then defined as

$$\hat{x}(k+1) = \hat{w}_s^{\mathrm{T}}(k)\varphi(\bar{z}(k)) - v(k) \tag{6}$$

where $\hat{x}(k)$ is the estimated system state vector, $\hat{w}_s(k)$ is the estimation of the ideal weight matrix w_s, and $v(k)$ is the robustifying term defined as

$$v(k) = \hat{\lambda}(k)\tilde{x}(k)/(\tilde{x}^{\mathrm{T}}(k)\tilde{x}(k) + C_s) \tag{7}$$

where $\tilde{x}(k) = x(k) - \hat{x}(k)$ is the system identification error, $\hat{\lambda}(k) \in \Re$ is an additional tunable parameter and $C_s > 0$ is a constant.

Next, the identification error dynamics are written as

$$\begin{aligned}\tilde{x}(k+1) &= x(k+1) - \hat{x}(k+1)\\ &= \tilde{w}_s^{\mathrm{T}}(k)\varphi(\bar{z}(k)) + \frac{\hat{\lambda}(k)\tilde{x}(k)}{\tilde{x}^{\mathrm{T}}(k)\tilde{x}(k) + C_s} + \varepsilon_s(k)\end{aligned} \tag{8}$$

where $\tilde{w}_s(k) = w_s - \hat{w}_s(k)$. Before proceeding, the following mild assumption is needed.

Assumption 1. The term $\varepsilon_s(k)$ is assumed to be upper bounded by a function of estimation error such that

$$\varepsilon_s^{\mathrm{T}}(k)\varepsilon_s(k) \leq \varepsilon_M(k) = \lambda^*\tilde{x}^{\mathrm{T}}(k)\tilde{x}(k) \tag{9}$$

where λ^* is the bounded constant target value such that $\|\lambda^*\| \leq \lambda_M$.

Remark 1. Eq. (9) corresponds to nonlinear small-gain type norm bounded uncertain characterization of $\varepsilon_s(k)$, which is used by a number of control researchers (Dierks, Thumati, & Jagannathan, 2009; Hayakawa et al., 2008) and is considered mild in comparison with the system uncertainty and the approximation errors bounded above by a known constant.

Moving on, adding and subtracting $\lambda^*\tilde{x}(k)/\tilde{x}^{\mathrm{T}}(k)\tilde{x}(k) + C_s$ to (8) allows the identification error dynamics to be rewritten as

$$\tilde{x}(k+1) = \Psi_1(k) + \Psi_2(k) + \lambda^*\tilde{x}(k)/(\tilde{x}^{\mathrm{T}}(k)\tilde{x}(k) + C_s) + \varepsilon_s(k) \tag{10}$$

where $\Psi_1(k) = \tilde{w}_s^{\mathrm{T}}(k)\varphi(\bar{z}(k))$ and $\Psi_2(k) = \tilde{\lambda}(k)\tilde{x}(k)/(\tilde{x}^{\mathrm{T}}(k)\tilde{x}(k) + C_s)$ with $\tilde{\lambda}(k) = \lambda^* - \hat{\lambda}(k)$.

The tuning laws for $\hat{w}_s(k)$ and $\hat{\lambda}(k)$ are now proposed as (Dierks et al., 2009)

$$\hat{w}_s(k+1) = \hat{w}_s(k) + \alpha_s\varphi(\bar{z}(k))\tilde{x}^{\mathrm{T}}(k+1) \tag{11}$$

and

$$\hat{\lambda}(k+1) = \hat{\lambda}(k) - \gamma_s\tilde{x}^{\mathrm{T}}(k+1)\tilde{x}(k)/(\tilde{x}^{\mathrm{T}}(k)\tilde{x}(k) + C_s), \tag{12}$$

respectively, where $\alpha_s > 0$ is the NN learning rate and $\gamma_s > 0$ is a design parameter. The update laws (11) and (12) are derived from Lyapunov theory in order to guarantee the local asymptotic stability of the proposed estimator scheme compared to boundedness results from other works (Jagannathan, 2006). Additionally, no prior offline training is needed for tuning the networks. Next, the stability analysis of the identification scheme is presented.

Theorem 1. *Let the proposed identification scheme in (6) be used to identify the system dynamics in (4), and let the parameter update law given in (11) and (12) be used for tuning the NN weights and the robust modification term respectively. In the presence of bounded uncertainties, the state estimation error $\tilde{x}(k)$ is asymptotically stable while the parameter estimation errors $\tilde{w}_s^{\mathrm{T}}(k)$ and $\tilde{\lambda}(k)$, respectively, are bounded.*

Proof. Consider the following positive definite Lyapunov function defined as

$$L(k) = \tilde{x}^{\mathrm{T}}(k)\tilde{x}(k) + \mathrm{tr}[\tilde{w}_s^{\mathrm{T}}(k)\tilde{w}_s(k)]/\alpha_s + \tilde{\lambda}^2(k)/\gamma_s \tag{13}$$

whose first difference is given by

$$\begin{aligned}\Delta L = &\underbrace{\tilde{x}^{\mathrm{T}}(k+1)\tilde{x}(k+1) - \tilde{x}^{\mathrm{T}}(k)\tilde{x}(k)}_{\Delta L_1} + \underbrace{\left(\tilde{\lambda}^2(k+1) - \tilde{\lambda}^2(k)\right)/\gamma_s}_{\Delta L_2}\\ &+ \underbrace{\mathrm{tr}[\tilde{w}_s^{\mathrm{T}}(k+1)\tilde{w}_s(k+1) - \tilde{w}_s^{\mathrm{T}}(k)\tilde{w}_s(k)]/\alpha_s}_{\Delta L_3}.\end{aligned}$$

Considering first ΔL_1 and substituting the identification error dynamics (10) as well as performing some mathematical manipulation reveals

$$\begin{aligned}\Delta L_1 = &\Psi_1^{\mathrm{T}}(k)\Psi_1(k) - 2\Psi_1^{\mathrm{T}}(k)\Psi_2(k) + \frac{2\lambda^*\Psi_1^{\mathrm{T}}(k)\tilde{x}(k)}{\tilde{x}^{\mathrm{T}}(k)\tilde{x}(k) + C_s}\\ &+ 2\Psi_1^{\mathrm{T}}(k)\varepsilon_s(k) + \Psi_2^{\mathrm{T}}(k)\Psi_2(k) - 2\Psi_2^{\mathrm{T}}(k)\varepsilon_s(k)\\ &+ \frac{2\lambda^*\varepsilon^{\mathrm{T}}(k)\tilde{x}(k)}{\tilde{x}^{\mathrm{T}}(k)\tilde{x}(k) + C_s} + \varepsilon_s^{\mathrm{T}}(k)\varepsilon_s(k) - \frac{2\lambda^*\Psi_2^{\mathrm{T}}(k)\tilde{x}(k)}{\tilde{x}^{\mathrm{T}}(k)\tilde{x}(k) + C_s}\\ &+ \frac{\lambda^{*2}\tilde{x}^{\mathrm{T}}(k)\tilde{x}(k)}{\left(\tilde{x}^{\mathrm{T}}(k)\tilde{x}(k) + C_s\right)^2} - \tilde{x}^{\mathrm{T}}(k)\tilde{x}(k).\end{aligned}$$

Next, considering ΔL_2 and substituting the parameter update law (12) reveals after manipulation

$$\begin{aligned}\Delta L_2 = &2\Psi_1^{\mathrm{T}}(k)\Psi_2(k) - 2\Psi_2^{\mathrm{T}}(k)\Psi_2(k) + \frac{2\lambda^*\Psi_2^{\mathrm{T}}(k)\tilde{x}(k)}{\tilde{x}^{\mathrm{T}}(k)\tilde{x}(k) + C_s}\\ &+ 2\varepsilon_s^{\mathrm{T}}(k)\Psi_2(k) + \gamma_s\frac{\left(\tilde{x}^{\mathrm{T}}(k+1)\tilde{x}(k)\right)^2}{\left(\tilde{x}^{\mathrm{T}}(k)\tilde{x}(k) + C_s\right)^2}.\end{aligned}$$

Now, considering ΔL_3 as well as substituting the NN weight tuning law (11), the error dynamics (10), as well as applying the Cauchy–Schwarz (C–S) inequality $((a_1+a_2+\cdots+a_n)^{\mathrm{T}}.(a_1+a_2+\cdots+a_n) \leq n.(a_1^{\mathrm{T}}a_1 + a_2^{\mathrm{T}}a_2 + \cdots + a_n^{\mathrm{T}}a_n))$ and applying the trace operator reveals

$$\begin{aligned}\Delta L_3 \leq\ &4\alpha_s\varphi(\bar{z}(k))^{\mathrm{T}}\varphi(\bar{z}(k)) \times \left(\Psi_1^{\mathrm{T}}(k)\Psi_1(k) + \Psi_2^{\mathrm{T}}(k)\Psi_2(k)\right.\\ &\left. + \frac{\lambda^{*2}\tilde{x}^{\mathrm{T}}(k)\tilde{x}(k)}{\left(\tilde{x}^{\mathrm{T}}(k)\tilde{x}(k) + C_s\right)^2} + \varepsilon_s^{\mathrm{T}}(k)\varepsilon_s(k)\right) - 2\Psi_1^{\mathrm{T}}(k)\\ &\times \left(\Psi_1(k) + \varepsilon_s(k) + \frac{\lambda^*\tilde{x}(k)}{\tilde{x}^{\mathrm{T}}(k)\tilde{x}(k) + C_s} - \Psi_2(k)\right).\end{aligned}$$

Recalling $\Delta L = \Delta L_1 + \Delta L_2 + \Delta L_3$, the first difference of (13) is rewritten as

$$\begin{aligned}\Delta L \leq\ &-(1-4\gamma_s)\Psi_1^{\mathrm{T}}(k)\Psi_1(k) - (1-4\gamma_s)\Psi_2^{\mathrm{T}}(k)\Psi_2(k)\\ &- \tilde{x}^{\mathrm{T}}(k)\tilde{x}(k) + 2\Psi_1^{\mathrm{T}}(k)\Psi_2(k) + 4\alpha_s\varphi(\bar{z}(k))^{\mathrm{T}}\varphi(\bar{z}(k))\Psi_1^{\mathrm{T}}(k)\Psi_1(k)\\ &+ 4\alpha_s\varphi(\bar{z}(k))^{\mathrm{T}}\varphi(\bar{z}(k))\Psi_2^{\mathrm{T}}(k)\Psi_2(k) + 4\alpha_s\varphi(\bar{z}(k))^{\mathrm{T}}\varphi(\bar{z}(k))\varepsilon_s^{\mathrm{T}}(k)\varepsilon_s(k)\\ &+ (2+4\gamma_s)\,\varepsilon^{\mathrm{T}}(k)\varepsilon(k) + \left(4\alpha_s\varphi(\bar{z}(k))^{\mathrm{T}}\varphi(\bar{z}(k)) + 2 + 4\gamma_s\right)\\ &\times \frac{\lambda^{*2}\tilde{x}^{\mathrm{T}}(k)\tilde{x}(k)}{\left(\tilde{x}^{\mathrm{T}}(k)\tilde{x}(k) + C_s\right)^2}\end{aligned}$$

after applying the C–S inequality twice. Now, taking the Euclidean norm while observing the facts $\|\lambda^*\| \leq \lambda_M$ (Assumption 1) and $\tilde{x}^{\mathrm{T}}(k)\tilde{x}(k)/\left(\tilde{x}^{\mathrm{T}}(k)\tilde{x}(k)+C_s\right)^2 < \tilde{x}^{\mathrm{T}}(k)\tilde{x}(k)$ yields

$$\begin{aligned}\Delta L \leq & -\left(1-2(\lambda_M+\lambda_M^2)-4(\alpha_s\varphi_M^2(\lambda_M+\lambda_M^2)\right.\\ & \left.+\ \gamma_s(\lambda_M+\lambda_M^2))\right)\left\|\tilde{x}(k)\right\|^2-(1-4\alpha_s\varphi_M^2-4\gamma_s)\left\|\Psi_1(k)\right\|^2\\ & -(1-4\alpha_s\varphi_M^2-4\gamma_s)\left\|\Psi_2(k)\right\|^2+2\left\|\Psi_1(k)\right\|\left\|\Psi_2(k)\right\|.\end{aligned}$$

In the next step, we define change of variables as $\bar{\Psi}_1(k) = \Psi_1(k)/\beta_1$ and $\bar{\Psi}_2(k) = \Psi_2(k)/\beta_2$ where β_1 and β_2 are constants. Additionally, the design parameters are selected as $\gamma_s = \alpha_s\varphi_M^2$, $\alpha_s \leq \beta_1^2/(8\varphi_M^2)$, and $\beta_2 = \delta/\beta_1$ where $\delta > 0$ is a constant. Using these relations and new variables, applying the C-S inequality, and combining like terms yields

$$\begin{aligned}\Delta L \leq & -\left(1-2(\lambda_M+\lambda_M^2)-\beta_1^2(\lambda_M+\lambda_M^2)\right)\left\|\tilde{x}(k)\right\|^2\\ & -(1-\beta_1^2-\delta/\beta_1^2)\left\|\tilde{w}_s^{\mathrm{T}}(k)\varphi(\bar{z}(k))\right\|^2-(1-\beta_1^2(1-1/\delta))\\ & \times\left\|\tilde{\lambda}(k)\right\|^2\left\|(\tilde{x}(k)/\tilde{x}^{\mathrm{T}}(k)\tilde{x}(k)+C_s)\right\|^2. \qquad (14)\end{aligned}$$

Therefore, $\Delta L \leq 0$ provided $\beta_1 \leq \min\{a_s, b_s, c_s, d_s\}$ and $\delta < 1/4$ where $a_s = \sqrt{(1+\sqrt{1-4\delta})/2}$, $b_s = \sqrt{(1-\sqrt{1-4\delta})/2}$, $c_s = \sqrt{\delta/(1+\delta)}$ and $d_s = \sqrt{(1-2(\lambda_M+\lambda_M^2))/(\lambda_M+\lambda_M^2)}$. Thus, $\tilde{x}(k)$,$\tilde{w}_s(k)$, and $\tilde{\lambda}(k)$ are bounded provided $\tilde{x}(k_0)$, $\tilde{w}_s(k_0)$, and $\tilde{\lambda}(k_0)$ are bounded in the compact set S. Further, by using Jagannathan (2006) and summing both sides of (14) to infinity and taking the limit as $k \to \infty$ reveals the asymptotic stability of estimation error $\left\|\tilde{x}(k)\right\|$; that is, $\left\|\tilde{x}(k)\right\|$ approaches zero as $k \to \infty$. □

Remark 2. The values of the constant parameters used in the proof of Theorem 1 are related to design parameters determined by the user and the inequalities derived above, and on the upper bounds relating to the NN reconstruction error and activation functions, λ_M and ϕ_M, respectively. In practice, the upper bound of the activation functions are easily determined, and the upper bound relating to the NN reconstruction error can be reduced by increasing the number of hidden layer neurons (Jagannathan, 2006).

Remark 3. According to the results of Theorem 1, the robustifying term of the online identifier (7) approaches zero after a sufficiently long online learning session, and the dynamic system (4) can be rewritten as

$$x(k+1) = \hat{x}(k+1) = \hat{w}_s^{\mathrm{T}}(k)\varphi(\bar{z}(k)). \qquad (15)$$

In the following section, discrete-time nonlinear optimal control for unknown affine systems will be achieved using only the result of the online system identification (15), but first, we present the following corollary.

Corollary 1 (*Dierks et al., 2009*). *Using the online system identification scheme (6), an estimate for $g(x(k))$ is given by*

$$g(x(k)) = \hat{w}_s^{\mathrm{T}}(k)\varphi'(\bar{z}(k))v_s^{\mathrm{T}}(\partial z(k)/\partial u(k)) \qquad (16)$$

where $\varphi'(\bar{z}(k)) \in \Re^{\ell x \ell}$ is the derivative of the activation function with respect to $\bar{z}(k)$ and $\partial z(k)/\partial u(k)$ is a constant matrix.

Proof. See (Dierks et al., 2009). □

4. Optimal control of unknown nonlinear discrete-time systems

Consider again the cost function (2). Using only the learned NN model (15), solving for the optimal control policy by setting $\partial V(x(k))/\partial u(k) = 0$ reveals

$$\begin{aligned}& 2Ru(k)+\left(\frac{\partial\hat{x}(k+1)}{\partial u(k)}\right)^{\mathrm{T}}\frac{\partial V(\hat{x}(k+1))}{\partial\hat{x}(k+1)}\\ & =2Ru(k)+\left(\hat{w}_s^{\mathrm{T}}(k)\varphi'(\bar{z}(k))v_s^{\mathrm{T}}\frac{\partial z(k)}{\partial u(k)}\right)^{\mathrm{T}}\frac{\partial V(\hat{x}(k+1))}{\partial\hat{x}(k+1)}=0. \qquad (17)\end{aligned}$$

Examining (17), it is clear that that an explicit expression for the optimal control $u^*(k)$ is not available. However, it will be shown in the following section that this deficiency does not prevent implementation of the HDP algorithm and that the optimal control of unknown nonlinear systems can still be obtained.

The optimal control policy that solves (17) ensures the equivalent system (15) is regulated in an optimal manner. On the other hand, since (15) is an equivalent representation of (4), optimal regulation of (15) ensures the optimal regulation of (4). In the following discussion of the HDP algorithm, i will denote the value iteration index while k is the time index.

To begin the algorithm, the cost function is initialized to a starting value which is commonly taken as $V_0(x(k)) = 0$, and the control policy $u_0(x(k))$ is selected as (Prokhorov & Wunsch, 1997)

$$u_0(\hat{x}(k)) = \arg\min_u\left(Q(x(k))+u(k)^{\mathrm{T}}Ru(k)+V_0(\hat{x}(k+1))\right).$$

After determining the admissible control policy $u_0(\hat{x}(k))$, the value function is then iterated according to

$$\begin{aligned}V_1(\hat{x}(k)) &= Q(x(k))+u_0^{\mathrm{T}}(k)Ru_0(k)+V_0\left(\hat{w}_s^{\mathrm{T}}(k)\varphi(\bar{z}(k))\right)\\ &= Q(x(k))+u_0^{\mathrm{T}}(k)Ru_0(k)+V_0(\hat{x}(k+1)).\end{aligned}$$

Continuing in this way, optimization is achieved by iterating between a sequence of control policies $u_i(\hat{x}(k))$calculated from

$$u_i(\hat{x}(k)) = \arg\min_u\left(Q(x(k))+u(k)^{\mathrm{T}}Ru(k)+V_i(\hat{x}(k+1))\right) \qquad (18)$$

and a sequence of positive value functions $V_i(\hat{x}(k))$ given by

$$V_{i+1}(\hat{x}(k)) = Q(x(k))+u_i^{\mathrm{T}}(k)Ru_i(k)+V_i\left(\hat{w}_s^{\mathrm{T}}(k)\varphi(\bar{z}(k))\right). \qquad (19)$$

Theorem 2. *Convergence of the HDP Algorithm: Consider the sequences $u_i(x(k))$ and $V_i(x(k))$ defined by (17) and (18), respectively. If $V_0(x(k)) = 0$, then it follows that V_i is a non-decreasing sequence for all i. Moreover, as $i \to \infty$, $V_i \to V_i^*$ and $u_i \to u_i^*$.*

As shown in Theorem 2 (Al-Tamimi et al., 2008), iterating between the sequences (18) and (19) guarantees that both sequences converge to the optimal cost function and optimal control policy, respectively, for the dynamic system (15), and thus (4) is also regulated in an optimal manner. Next, the NN implementation of the HDP algorithm is discussed along with the proof of convergence.

5. HDP algorithm design using neural networks

Even when the complete dynamics of the nonlinear system are known, the optimal control policy (3) and cost function (2) are difficult to calculate. Thus, to implement the HDP algorithm, the learning capabilities of NN have been widely used (Si et al., 2004); however, an inherent assumption of most implementations of the HDP algorithm require at least partial knowledge of the system dynamics as well as an expression for the controlled system one step ahead (Si et al., 2004). In the following development, explicit knowledge of the system dynamics $f(x(k))$ and $g(x(k))$ are not required. Only the NN representation (15) will be used in the implementation of the HDP algorithm.

5.1. Approximation of the optimal value function and optimal control policy using NN

Using the *universal approximation property* of NN (Jagannathan, 2006), the value function (19) and control action (18) have a NN representation written as

$$V_i(x(k)) = W_{Vi}^{\mathrm{T}}\sigma(x(k)) + \varepsilon_{Vi} \tag{20}$$

and

$$u_i(x(k)) = W_{Ai}^{\mathrm{T}}\vartheta(x(k)) + \varepsilon_{Ai} \tag{21}$$

respectively, where W_{Vi} and W_{Ai} are the bounded ideal NN weights and ε_{Vi} and ε_{Ai} are the bounded approximation errors for the critic and action networks, respectively. The upper bounds for the ideal NN weights are taken as $\|W_{Vi}\| \leq W_{VM}$ and $\|W_{Ai}\|_F \leq W_{AM}$ while the approximation errors are upper bounded as $\|\varepsilon_{Vi}\| \leq \varepsilon_{VM}$ and $\|\varepsilon_{Ai}\| \leq \varepsilon_{AM}$, where W_{VM}, W_{AM}, ε_{VM} and ε_{AM} are positive constants (Jagannathan, 2006). Additionally, in this work it will be assumed that $\left\|\partial\varepsilon_{Vi}/(\partial\hat{x}(k+1))\right\| \leq \varepsilon'_{VM}$ where ε'_{VM} is another positive constant. The NN representations in (20) and (21) are linear in the tunable parameters, and this type of NN topology is common in the nonlinear ADP literature (Al-Tamimi et al., 2008; Chen & Jagannathan, 2008).

The NN approximations of (20) and (21) are now written as (Al-Tamimi et al., 2008; Dierks et al., 2009)

$$\hat{V}_i(x(k)) = \hat{W}_{Vi}^{\mathrm{T}}\sigma(x(k)) = \hat{W}_{Vi}^{\mathrm{T}}\sigma(k) \tag{22}$$

and

$$\hat{u}_i(x(k)) = \hat{W}_{Ai}^{\mathrm{T}}\vartheta(x(k)) = \hat{W}_{Ai}^{\mathrm{T}}\vartheta(k) \tag{23}$$

where $\hat{W}_{Vi}$ and $\hat{W}_{Ai}$ are the approximation of the ideal weights W_{Vi} and W_{Ai}, respectively. The activation functions $\sigma(\bullet)$ and $\vartheta(\bullet)$ are each chosen to be basis sets and thus are linearly independent. Further, recalling $V_i(0) = 0$ and $u_i(0) = 0$, the basis functions are selected so that $\sigma(0) = 0$ and $\vartheta(0) = 0$, respectively. In this work, the basis function $\sigma(\bullet)$ is obtained from an even polynomial so that the positive definite value function can be approximated. Additionally, the basis function for $\vartheta(\bullet)$ is selected from the derivative of the critic basis function $\sigma(\bullet)$ since the optimal control (3) is dependent on the gradient of the critic network (Al-Tamimi et al., 2008; Chen & Jagannathan, 2008).

The critic weights are tuned at each iteration of the HDP algorithm to minimize the residual error $\hat{V}_{i+1}(x(k)) - d_i(x(k), \hat{x}(k+1), W_{Vi}, W_{Ai})$ in a least squares sense for a set of states $x(k)$ sampled from a compact set $\Omega \in \Re^n$ where $d_i(x(k), \hat{x}(k+1), W_{Vi}, W_{Ai}) = Q(x(k)) + \hat{u}_i^{\mathrm{T}}(k)R\hat{u}_i(k) + V_i\left(\hat{w}_s^{\mathrm{T}}(k)\varphi(\bar{z}(k))\right)$. The critic error then becomes

$$e_V(x) = \hat{W}_{Vi+1}^{\mathrm{T}}\sigma(x(k)) - d_i(x(k), \hat{x}(k+1), W_{Vi}, W_{Ai}), \tag{24}$$

and it is observed that the critic error (24) is linear in the tunable weights, $\hat{W}_{Vi+1}$. Therefore, the weight update law is found using the method of weighted residuals, and is determined by projecting the critic error on to the partial derivative of the critic error with respect to $\hat{W}_{Vi+1}$ and setting the result to zero (Al-Tamimi et al., 2008; Chen & Jagannathan, 2008). Applying this method reveals

$$\langle \partial e_V(x)/\partial W_{Vi+1}, e_V(x)\rangle = 0 \tag{25}$$

where $\langle f, g\rangle = \int_\Omega fg^{\mathrm{T}}\mathrm{d}x$, and the unique solution for $\hat{W}_{Vi+1}$ is found to be

$$\hat{W}_{Vi+1} = \left(\int_\Omega \sigma(x(k))\sigma^{\mathrm{T}}(x(k))\mathrm{d}x\right)^{-1} \times \int_\Omega \sigma(x(k))d_i^{\mathrm{T}}(x(k), \hat{x}(k+1), W_{Vi}, W_{Ai})\mathrm{d}x. \tag{26}$$

Remark 4. The activation function $\sigma(\bullet)$ must be chosen to be linearly independent on the compact set Ω to ensure that the weight update law (26) is valid and a unique solution for $\hat{W}_{Vi+1}$ exists. This requirement also guarantees that $\int_\Omega \sigma(x(k))\sigma^{\mathrm{T}}(x(k))\mathrm{d}x$ is invertible (Chen & Jagannathan, 2008).

According to (18), at each iteration index the optimal control policy that minimizes the cost function is found. Using the NN implementation, this is equivalent to finding the optimal weight matrix W_{Ai} that minimizes the cost function; however, an explicit solution for W_{Ai} is difficult to find at each iteration. As a result, a second training loop is introduced to find the action network weights that minimize the cost function (19), and the index for this loop will be denoted by j.

To begin the development of the second loop, we define the action error to be the difference between the estimated control input $\hat{u}_{ij}(k)$ in (23) and the control input that minimizes the estimated cost function $\hat{u}_i^*(k)$. The action error is written as

$$\begin{aligned} e_{ai}(j) &= \hat{u}_i(k) - \hat{u}_i^*(k) \\ &= \hat{W}_{Ai}^{\mathrm{T}}(j)\vartheta(x(k)) + \frac{1}{2}R^{-1}g^{\mathrm{T}}(x(k))\frac{\partial\hat{V}_i(x_j(k+1))}{\partial x_j(k+1)}. \end{aligned} \tag{27}$$

Next, we utilize (22) as well as the results of Theorem 1 and Corollary 1 to rewrite (27) as

$$\begin{aligned} e_{ai}(j) &= \hat{W}_{Ai}^{\mathrm{T}}(j)\vartheta(x(k)) + \frac{1}{2}R^{-1}\left(\frac{\partial z_j(k)}{\partial\hat{u}_{ij}(k)}\right)^{\mathrm{T}} \\ &\quad \times v_s\varphi'(\bar{z}_j(k))^{\mathrm{T}}\hat{w}_s(k)\frac{\partial\sigma(\hat{x}_j(k+1))}{\partial\hat{x}_j(k+1)}\hat{W}_{Vi}. \end{aligned} \tag{28}$$

The dynamics of (28) with respect to the index j are now written as

$$\begin{aligned} e_{ai}(j+1) &= \hat{W}_{Ai}^{\mathrm{T}}(j+1)\vartheta(x(k)) + \frac{1}{2}R^{-1}\left(\frac{\partial z_{j+1}(k)}{\partial\hat{u}_{ij+1}(k)}\right)^{\mathrm{T}} \\ &\quad \times v_s\varphi'(\bar{z}_{j+1}(k))^{\mathrm{T}}\hat{w}_s(k)\frac{\partial\sigma(\hat{x}_{j+1}(k+1))}{\partial\hat{x}_{j+1}(k+1)}\hat{W}_{Vi}. \end{aligned} \tag{29}$$

For convenience in the stability proof in the following section, the weight update for the action NN is selected to be

$$\hat{W}_{Ai}(j+1) = \hat{W}_{Ai}(j) - \alpha_j\vartheta(k)e_{ai}^{\mathrm{T}}(j)/(\vartheta^{\mathrm{T}}(k)\vartheta(k)+1) \tag{30}$$

where α_j is a small positive design parameter and varies with the index j. The weight update law (30) can be viewed as a control input for the action error (29). Next, defining the weight estimation errors $\tilde{W}_{Ai}(j) = W_{Ai} - \hat{W}_{Ai}(j)$ and $\tilde{W}_{Vi} = W_{Vi} - \hat{W}_{Vi}$ as well as substituting the tuning law (30) into (29) reveals the closed loop error system to be

$$\begin{aligned} e_{ai}(j+1) &= -\tilde{W}_{Ai}(j)\vartheta(x(k)) - e_{ai}(j)\left(\frac{\alpha_j\vartheta^{\mathrm{T}}(k)\vartheta(x(k))}{\vartheta^{\mathrm{T}}(k)\vartheta(k)+1}\right) \\ &\quad -\frac{1}{2}R^{-1}\left(\frac{\partial z_{j+1}(k)}{\partial\hat{u}_{ij+1}(k)}\right)^{\mathrm{T}}v_s\varphi'(\bar{z}_{j+1}(k))^{\mathrm{T}}\hat{w}_s(k)\frac{\partial\sigma(\hat{x}_{j+1}(k+1))}{\partial\hat{x}_{j+1}(k+1)}\tilde{W}_{Vi} \\ &\quad +W_{Ai}\vartheta(x(k)) + \frac{1}{2}R^{-1}\left(\frac{\partial z_{j+1}(k)}{\partial\hat{u}_{ij+1}(k)}\right)^{\mathrm{T}}v_s\varphi'(\bar{z}_{j+1}(k))^{\mathrm{T}}\hat{w}_s(k) \\ &\quad \times\frac{\partial\sigma(\hat{x}_{j+1}(k+1))}{\partial\hat{x}_{j+1}(k+1)}W_{Vi}. \end{aligned} \tag{31}$$

Now, we utilize the fact that $u_i(x(k))$ in (21) minimizes the cost function (20) revealing

$$0 = \varepsilon_{Ai}(j) + \frac{1}{2}R^{-1}\left(\frac{\partial z_{j+1}(k)}{\partial\hat{u}_{ij+1}(k)}\right)^{\mathrm{T}}v_s\varphi'(\bar{z}_{j+1}(k))^{\mathrm{T}}\hat{w}_s(k)\frac{\partial\varepsilon_{Vi}}{\partial x_j(k+1)}$$

$$+ W_{Ai}^{\mathrm{T}}\vartheta(x(k)) + \frac{1}{2}R^{-1}\left(\frac{\partial z_{j+1}(k)}{\partial \hat{u}_{ij+1}(k)}\right)^{\mathrm{T}} v_s\varphi'(\bar{z}_{j+1}(k))^{\mathrm{T}}\hat{w}_s(k)$$
$$\frac{\partial\sigma(x_j(k+1))}{\partial x_j(k+1)}W_{Vi}$$

and the dynamics (31) can be rewritten as

$$e_{ai}(j+1) = -\tilde{W}_{Ai}(j)\vartheta(x(k)) - e_{ai}(j)\left(\frac{\alpha_j\vartheta^{\mathrm{T}}(k)\vartheta(x(k))}{\vartheta^{\mathrm{T}}(k)\vartheta(k)+1}\right)$$
$$-\frac{1}{2}R^{-1}\left(\frac{\partial z_{j+1}(k)}{\partial \hat{u}_{i(j+1)}(k)}\right)^{\mathrm{T}} v_s\varphi'(\bar{z}_{j+1}(k))^{\mathrm{T}}\hat{w}_s(k)$$
$$\times\frac{\partial\sigma(\hat{x}_{j+1}(k+1))}{\partial\hat{x}_{j+1}(k+1)}\tilde{W}_{Vi} - \varepsilon_{Ai}(j) - \frac{1}{2}R^{-1}$$
$$\times\left(\frac{\partial z_{j+1}(k)}{\partial \hat{u}_{i(j+1)}(k)}\right)^{\mathrm{T}} v_s\varphi'(\bar{z}_{j+1}(k))^{\mathrm{T}}\hat{w}_s(k)\frac{\partial\varepsilon_{Vi}}{\partial x_j(k+1)}. \tag{32}$$

As a final step, we write the estimation error dynamics of the action NN weights as

$$\tilde{W}_{Ai}(j+1) = \tilde{W}_{Ai} + \alpha_j\vartheta(k)e_{ai}^{\mathrm{T}}(j)/(\vartheta^{\mathrm{T}}(k)\vartheta(k)+1). \tag{33}$$

Fig. 1 displays a flow chart of the proposed scheme. First, the proposed NN identifier (6) is utilized to learn the nonlinear system dynamics (1) online. After the identification error has converged to zero (Theorem 1), the offline HDP training begins using NNs to learn the cost function and optimal control policy (22) and (23), respectively. Next, the stability of action network is investigated.

5.2. Proof of convergence for the action network

In the following theorem, the convergence of the action NN weights is demonstrated while explicitly considering the NN reconstruction errors ε_{Vi} and ε_{Ai}. Subsequently, the stability of the system is re-examined while ignoring the NN reconstruction errors similarly to previous works.

Theorem 3. *Convergence of the Action Network (Dierks et al., 2009): Given the error system for the action network given by (29), let the update law for the action NN weights be given by (30). Then, there exists a decreasing, positive definite sequence $J\left(e_{ei}(j), \tilde{W}_{Ai}(j)\right)$ such that the action network error (27) and the action network weight estimation errors $\tilde{W}_{Ai}(j)$ converge uniformly to a bounded region near the origin as $j \to \infty$.*

Proof. Consider the positive definite sequence defined by $J = J_1 + 5J_2$ where

$$J_1 = \alpha_j e_{ai}^{\mathrm{T}}(j)e_{ai}(j)/\Lambda(j) \tag{34}$$

with $\Lambda(j) = \left(\vartheta^{\mathrm{T}}(k)\vartheta(k)+1\right)\left(\left\|\partial\sigma(\hat{x}_j(k+1))/\partial\hat{x}_j(k+1)\right\|_F^2 + 1\right) > 0$ and

$$J_2 = \mathrm{tr}\{\tilde{W}_{Ai}^{\mathrm{T}}(j)\tilde{W}_{Ai}(j)\}. \tag{35}$$

The first difference of J is given by $J = J(j+1) - J(j) = \Delta J_1 + 5\Delta J_2$, and first, we consider ΔJ_1. Taking the first difference of J_1 along with substituting the closed loop action error dynamics (32) reveals

$$\Delta J_1 = -\frac{\alpha_j e_{ai}^{\mathrm{T}}(j)e_{ai}(j)}{\Lambda(j)} + \frac{\alpha_{j+1}}{\Lambda(j+1)}$$
$$\times\left(-\vartheta^{\mathrm{T}}(k)\tilde{W}_{AI} - \frac{1}{2}\tilde{W}_{Vi}^{\mathrm{T}}\frac{\partial\sigma(\hat{x}_{j+1}(k+1))}{\partial\hat{x}_{j+1}(k+1)}^{\mathrm{T}}\hat{w}_s^{\mathrm{T}}(k)\varphi'(\bar{z}_{j+1}(k))\right.$$
$$\left.\times v_s^{\mathrm{T}}\left(\frac{\partial z_{j+1}(k)}{\partial\hat{u}_{i(j+1)}(k)}\right)R^{-T} - \varepsilon_A^{\mathrm{T}}(j) - \frac{1}{2}\frac{\partial\varepsilon_V}{\partial\hat{x}(k+1)}^{\mathrm{T}}\hat{w}_s^{\mathrm{T}}(k)\varphi'(\bar{z}(k))_{j+1}\right.$$
$$\left.\times v_s^{\mathrm{T}}\left(\frac{\partial z_{j+1}(k)}{\partial\hat{u}_{i(j+1)}(k)}\right)R^{-T} - \left(\frac{\alpha_j\vartheta^{\mathrm{T}}(k)\vartheta(k)}{\vartheta^{\mathrm{T}}(k)\vartheta(k)+1}\right)^{\mathrm{T}}e_{ai}^{\mathrm{T}}(j)\right)$$
$$\times\left(-\tilde{W}_{AI}^{\mathrm{T}}\vartheta(k) - \frac{1}{2}R^{-1}\left(\frac{\partial z_{j+1}(k)}{\partial\hat{u}_{i(j+1)}(k)}\right)^{\mathrm{T}}v_s\varphi'(\bar{z}_{j+1}(k))^{\mathrm{T}}\hat{w}_s(k)\right.$$
$$\left.\times\frac{\partial\sigma(\hat{x}_{j+1}(k+1))}{\partial\hat{x}_{j+1}(k+1)}\tilde{W}_{Vi} - \varepsilon_A(j) - \frac{1}{2}R^{-1}\left(\frac{\partial z_{j+1}(k)}{\partial\hat{u}_{i(j+1)}(k)}\right)^{\mathrm{T}}\right.$$
$$\left.\times v_s\varphi'(\bar{z}_{j+1}(k))^{\mathrm{T}}\hat{w}_s(k)\frac{\partial\varepsilon_V}{\partial\hat{x}_j(k+1)} - e_{ai}\left((j)\left(\frac{\alpha_j\vartheta^{\mathrm{T}}(k)\vartheta(k)}{\vartheta^{\mathrm{T}}(k)\vartheta(k)+1}\right)\right)\right).$$

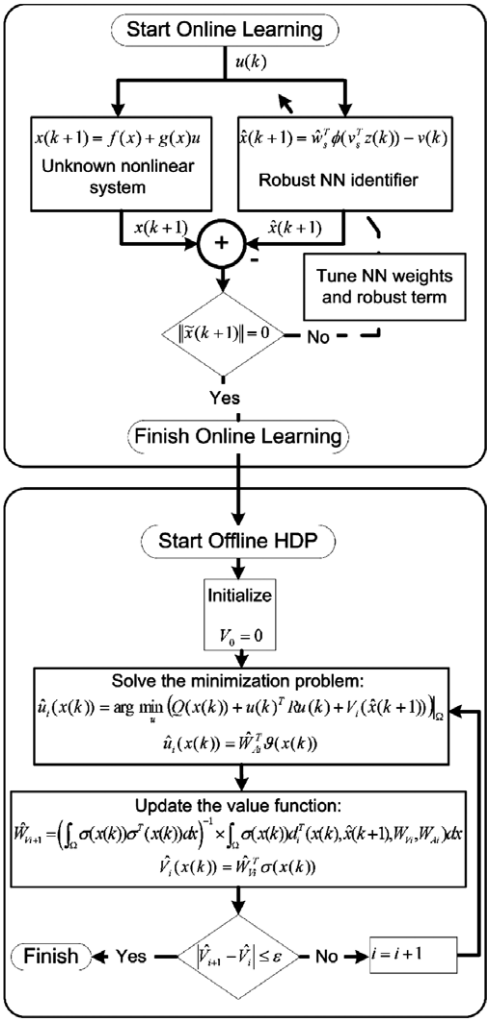

Fig. 1. Flow chart of the proposed scheme.

Next, defining $\Xi_1^{\mathrm{T}} = \tilde{W}_{Ai}^{\mathrm{T}}\vartheta(k)$, applying the C–S inequality as well as taking the upper bound of ΔJ_1 reveals

$$\Delta J_1 \le -\frac{\alpha_j\|e_{ai}(j)\|^2}{\Lambda_j} + \frac{5\alpha_{j+1}\alpha_j^2}{\Lambda_{j+1}}\left(\frac{\vartheta^{\mathrm{T}}(k)\vartheta(k)}{\vartheta^{\mathrm{T}}(k)\vartheta(k)+1}\right)^2$$
$$\times\|e_{ai}(j)\|^2 + \frac{5\alpha_{j+1}}{\Lambda_{j+1}}\left(\|\Xi_1\|^2 + \varepsilon_{AM}^2\right.$$
$$\left.+\frac{1}{4}\Gamma^2\left\|\frac{\partial\sigma(\hat{x}_{j+1}(k+1))}{\partial\hat{x}_{j+1}(k+1)}\right\|_F^2\left\|\tilde{W}_{Vi}\right\|^2 + \frac{1}{4}\varepsilon_{vM}'^2\Gamma^2\right) \tag{36}$$

where $\Gamma^2 = \left\|\partial z_{j+1}(k)/\partial\hat{u}_{i(j+1)}(k)\right\|_F^2\left\|R^{-1}\right\|_F^2 v_{sM}^2\varphi_M'^2\hat{w}_{sM}^2$ is a computable constant with v_{sM}, $\hat{w}_{sM}$, and φ_M' being the upper

bounds for the identification network parameters v_s, $\hat{w}_s$, and $\varphi'(\bar{z}_{j+1}(k))$, respectively.

Now, taking the first difference of J_2 and substituting the weight estimation error dynamics for the action network (33) reveals

$$\Delta J_2 = \frac{\alpha_j^2 \vartheta^{\mathrm{T}}(k)\vartheta(k)}{\left(\vartheta^{\mathrm{T}}(k)\vartheta(k)+1\right)^2} \mathrm{tr}\left\{e_{ai}(j)e_{ai}^{\mathrm{T}}(j)\right\} + \mathrm{tr}\left\{2\tilde{W}_{Ai}^{\mathrm{T}}\left(\frac{\alpha_j \vartheta(k)}{\vartheta^{\mathrm{T}}(k)\vartheta(k)+1}\right)e_{ai}^{\mathrm{T}}(j)\right\}.$$

Next, we expand $e_{ai}^{\mathrm{T}}(j)$ in the second term of ΔJ_2 using (29) and similar steps as those used to form (32), use the previously defined variables Ξ_1 and Γ, and take the upper bound of ΔJ_2 to yield

$$\Delta J_2 \leq \frac{\left(\varepsilon_{AM}^2 + (1/2)\Gamma^2\left(\varepsilon_{VM}'^2 + W_{VM}^2 + 1\right)\right)}{\vartheta^{\mathrm{T}}(k)\vartheta(k)+1} + \frac{\alpha_j^2 \vartheta(k)^{\mathrm{T}}\vartheta(k)}{\left(\vartheta^{\mathrm{T}}(k)\vartheta(k)+1\right)^2}\|e_{ai}\|^2 - \frac{\alpha_j \|\Xi_1\|^2}{\vartheta^{\mathrm{T}}(k)\vartheta(k)+1} \times \left(2 - \alpha_j\left(\frac{3}{2} + \frac{1}{2}\left\|\left(\frac{\partial\sigma(\hat{x}(k+1))}{\partial\hat{x}(k+1)}\right)_j\right\|^2\left(1 + \left\|\hat{W}_{Vi}\right\|^2\right)\right)\right) \quad (37)$$

where the relation $\left\|\tilde{W}_{Vi}\right\| \leq W_{VM} + \left\|\hat{W}_{Vi}\right\|$ was utilized.

Finally, using (36) and (37), $\Delta J = \Delta J_1 + 5\Delta J_2$ is found to be the expression in Box I.

The first two terms of ΔJ are guaranteed to be less than zero provided the design parameter α_j is selected according to the expression in Box II.

Thus, it can be concluded that ΔJ is negative outside of a compact set revealing the sequence J to be a decreasing sequence. Since $J > 0$ and $\Delta J \leq 0$ outside of a compact set, it can be concluded that the action network error (27) and the action network weight estimation errors $\tilde{W}_{Ai}(j)$ converge uniformly to a bounded region near the origin. The bounds for the errors are given by the expressions in Box III. □

Remark 5. For the index j, $\left\|\tilde{W}_{Vi}\right\|^2$ is constant since $\hat{W}_{Vi}$ is tuned at each iteration i not j, but as i increases, $\left\|\tilde{W}_{vi}\right\|^2$ gets smaller using the least squares tuning law (26). Therefore, the upper bound on η decreases as i increases. This result makes sense because the action network is tuned using the critic network information. Similar to the results of Theorem 1, the values of the constant parameters used in the proof of Theorem 3 are related to design parameters determined by the user and the inequalities derived above, and on the upper bounds relating to the NN reconstruction error and activation functions. Additionally, the upper bounds of the gradient of the activation functions are easily determined since the activation functions are known before hand (Jagannathan, 2006).

Next, the stability of the action network is demonstrated under the assumption that the NN reconstruction errors ε_{Vi} and ε_{Ai} are negligible similarly to the works of Al-Tamimi et al. (2008), Chen and Jagannathan (2008).

Corollary 2. *Ideal Action Network Stability: Let the hypothesis of Theorem 3 hold in the absence of NN reconstruction errors. Then, there exists a decreasing, positive definite sequence* $J^*\left(e_{ei}(j), \tilde{W}_{Ai}(j)\right)$ *such that the action network error (27) and the action network weight estimation errors* $\tilde{W}_{Ai}(j)$ *converge to zero uniformly as* $j \to \infty$.

Proof. Consider the positive definite sequence $J^* = J_1^* + 3J_2^*$ where J_1^* and J_2^* are defined similarly to (34) and (35), respectively. As in the proof of Theorem 3, we take the first difference of J^* and first consider ΔJ_1^*. Applying similar steps as those used to derive (36), ΔJ_1^* in the absence of NN reconstruction errors is found to be

$$\Delta J_1^* \leq -\frac{\alpha_j \|e_{ai}(j)\|^2}{\Lambda_j} + \frac{3\alpha_{j+1}\alpha_j^2}{\Lambda_{j+1}}\left(\frac{\vartheta^{\mathrm{T}}(k)\vartheta(k)}{\vartheta^{\mathrm{T}}(k)\vartheta(k)+1}\right)^2\|e_{ai}(j)\|^2 + \frac{3\alpha_{j+1}}{\Lambda_{j+1}}\left(\|\Xi_1\|^2 + \frac{1}{4}\Gamma^2\left\|\frac{\partial\sigma(\hat{x}_{j+1}(k+1))}{\partial\hat{x}_{j+1}(k+1)}\right\|_F^2\left\|\tilde{W}_{Vi}\right\|^2\right). \quad (38)$$

Now, considering ΔJ_2^* and applying similar steps as those used to form (37) reveals

$$\Delta J_2^* \leq -\frac{\alpha_j \|\Xi_1\|^2}{\vartheta^{\mathrm{T}}(k)\vartheta(k)+1}\left(2 - \frac{\alpha_j}{2}\left\|\left(\frac{\partial\sigma(\hat{x}(k+1))}{\partial\hat{x}(k+1)}\right)_j^{\mathrm{T}}\right\|^2\right) + \frac{\alpha_j^2\vartheta(k)^{\mathrm{T}}\vartheta(k)}{\left(\vartheta^{\mathrm{T}}(k)\vartheta(k)+1\right)^2}\|e_{ai}\|^2 + \frac{1}{\vartheta^{\mathrm{T}}(k)\vartheta(k)+1}\left(\frac{\Gamma^2}{2}\left\|\tilde{W}_{vi}^{\mathrm{T}}\right\|^2\right). \quad (39)$$

Finally, we observe that as $i \to \infty$, $\left\|\tilde{W}_{vi}\right\|^2 \to 0$ using the least squares update (26). Thus, as $i \to \infty$, (38) and (39) are used to form $\Delta J^* = \Delta J_1^* + 3\Delta J_2^*$ written as

$$\Delta J^* \leq -\alpha_j\|e_{ai}(j)\|^2\left(\frac{1}{\Lambda_j} - \frac{3\alpha_{j+1}\alpha_j}{\Lambda_{j+1}}\left(\frac{\vartheta^{\mathrm{T}}(k)\vartheta(k)}{\vartheta^{\mathrm{T}}(k)\vartheta(k)+1}\right)^2 - \frac{3\alpha_j\vartheta(k)^{\mathrm{T}}\vartheta(k)}{\left(\vartheta^{\mathrm{T}}(k)\vartheta(k)+1\right)^2}\right) - \frac{3\alpha_j\|\Xi_1\|^2}{\vartheta^{\mathrm{T}}(k)\vartheta(k)+1} \times \left(2 - \frac{\alpha_j}{2}\left\|\left(\frac{\partial\sigma(\hat{x}(k+1))}{\partial\hat{x}(k+1)}\right)_j^{\mathrm{T}}\right\|^2 - \frac{\alpha_{j+1}\left(\vartheta^{\mathrm{T}}(k)\vartheta(k)+1\right)}{\alpha_j\Lambda_{j+1}}\right)$$

and $\Delta J^* \leq 0$ provided

$$\alpha_j < \min\left\{1/6\left(\left\|\frac{\partial\sigma(\hat{x}_j(k+1))}{\partial\hat{x}_j(k+1)}\right\|_F^2 + 1\right), 2\alpha_{j-1} - \frac{\alpha_{j-1}^2}{2} \times \left\|\left(\frac{\partial\sigma(\hat{x}_{j-1}(k+1))}{\partial\hat{x}_{j-1}(k+1)}\right)^{\mathrm{T}}\right\|^2, 4\Big/\left\|\left(\frac{\partial\sigma(\hat{x}_j(k+1))}{\partial\hat{x}_j(k+1)}\right)^{\mathrm{T}}\right\|^2\right\}.$$

Therefore, since $J^* > 0$ for all j and $\Delta J^* \leq 0$ as $i \to \infty$, it can be concluded that $\|e_{ai}\| \to 0$ and $\|\Xi_1\| \to 0$ as $j \to \infty$ revealing the action network errors converge to zero while the action network weights remain bounded in the absence of NN reconstruction errors. □

Remark 6. The results of Theorem 3 indicate that only a *nearly* optimal control law is possible when the NN reconstruction errors are explicitly considered. However, as illustrated by Corollary 2, the NN estimate of the optimal control law converges to the real optimal control uniformly when the NN reconstruction errors are ignored. Although the NN reconstruction errors are often ignored in the literature (Al-Tamimi et al., 2008; Chen & Jagannathan, 2008), proof of convergence for the NN implementations of HDP (22) and (23) are rarely studied.

6. Simulation results

To demonstrate the effectiveness of the proposed approach, the algorithm developed in this work was first implemented on

$$\Delta J \leq -\frac{5\alpha_j \|e_{ai}\|^2}{\left(\vartheta^{\mathrm{T}}(k)\vartheta(k)+1\right)}\left(\frac{1}{5\left(\left\|\frac{\partial\sigma(\hat{x}_j(k+1))}{\partial\hat{x}_j(k+1)}\right\|_F^2+1\right)}-\frac{\alpha_j\alpha_{j+1}\left(\frac{\vartheta^{\mathrm{T}}(k)\vartheta(k)}{\vartheta^{\mathrm{T}}(k)\vartheta(k)+1}\right)^2}{\left(\left\|\frac{\partial\sigma(\hat{x}_{j+1}(k+1))}{\partial\hat{x}_{j+1}(k+1)}\right\|_F^2+1\right)}-\frac{\alpha_j\vartheta(k)^{\mathrm{T}}\vartheta(k)}{\left(\vartheta^{\mathrm{T}}(k)\vartheta(k)+1\right)}\right)$$
$$-\frac{5\|\Xi_1\|^2}{\vartheta^{\mathrm{T}}(k)\vartheta(k)+1}\left(2\alpha_j-\alpha_j^2\left(\frac{3}{2}+\frac{1}{2}\left\|\left(\frac{\partial\sigma(\hat{x}_j(k+1))}{\partial\hat{x}_j(k+1)}\right)\right\|^2\left(1+\left\|\hat{W}_{vi}\right\|^2\right)\right)-\frac{\alpha_{j+1}}{\left(\left\|\frac{\partial\sigma(\hat{x}_{j+1}(k+1))}{\partial\hat{x}_{j+1}(k+1)}\right\|_F^2+1\right)}\right)+5\eta$$

where

$$\eta=\varepsilon_{AM}^2+(1/2)\Gamma^2\left(\varepsilon_{VM}'^2+W_{vM}^2+1\right)+\alpha_{j+1}$$
$$\times\left(\varepsilon_{AM}^2+\frac{1}{4}\Gamma^2\left\|\frac{\partial\sigma(\hat{x}_{j+1}(k+1))}{\partial\hat{x}_{j+1}(k+1)}\right\|_F^2\left\|\tilde{W}_{vi}\right\|^2+\frac{1}{4}\varepsilon_{vM}'^2\Gamma^2\right)\Big/\left(\left\|\frac{\partial\sigma(\hat{x}_{j+1}(k+1))}{\partial\hat{x}_{j+1}(k+1)}\right\|_F^2+1\right).$$

Box I.

$$\alpha_j<\min\left\{\frac{1}{10\left(\left\|\frac{\partial\sigma(\hat{x}_j(k+1))}{\partial\hat{x}_j(k+1)}\right\|_F^2+1\right)},\ \alpha_{j-1}\left(2-\alpha_{j-1}\left(\frac{3}{2}+\frac{1}{2}\left\|\left(\frac{\partial\sigma(\hat{x}_{j-1}(k+1))}{\partial\hat{x}_{j-1}(k+1)}\right)\right\|^2\left(1+\left\|\hat{W}_{vi}\right\|^2\right)\right)\right),\right.$$
$$\left.\frac{2}{\left(\frac{3}{2}+\frac{1}{2}\left\|\frac{\partial\sigma(\hat{x}_j(k+1))}{\partial\hat{x}_j(k+1)}\right\|_F^2\left(1+\left\|\hat{W}_{vi}\right\|^2\right)\right)}\right\}.$$

Box II.

$$\|e_{ai}\|<\sqrt{\|\eta\|\left(\vartheta^{\mathrm{T}}(k)\vartheta(k)+1\right)\Big/\left(\frac{\alpha_j}{5\left(\left\|\frac{\partial\sigma(\hat{x}_j(k+1))}{\partial\hat{x}_j(k+1)}\right\|_F^2+1\right)}-\frac{\alpha_j^2\alpha_{j+1}\left(\frac{\vartheta^{\mathrm{T}}(k)\vartheta(k)}{\vartheta^{\mathrm{T}}(k)\vartheta(k)+1}\right)^2}{\left(\left\|\frac{\partial\sigma(\hat{x}_{j+1}(k+1))}{\partial\hat{x}_{j+1}(k+1)}\right\|_F^2+1\right)}-\frac{\alpha_j^2\vartheta(k)^{\mathrm{T}}\vartheta(k)}{\left(\vartheta^{\mathrm{T}}(k)\vartheta(k)+1\right)}\right)}$$
$$\|\Xi_1\|<\sqrt{\|\eta\|\left(\vartheta^{\mathrm{T}}(k)\vartheta(k)+1\right)\Big/\left(2\alpha_j-\alpha_j^2\left(\frac{3}{2}+\frac{1}{2}\left\|\frac{\partial\sigma(\hat{x}_j(k+1))}{\partial\hat{x}_j(k+1)}\right\|_F^{2^2}\left(1+\left\|\hat{W}_{vi}\right\|^2\right)\right)-\frac{\alpha_{j+1}}{\left(\left\|\frac{\partial\sigma(\hat{x}_{j+1}(k+1))}{\partial\hat{x}_{j+1}(k+1)}\right\|_F^2+1\right)}\right)}.$$

Box III.

a linear system since the results can be easily verified by solving the discrete-time algebraic Riccati equation (DARE) and then on a nonlinear system. Due to space constraints, the linear system example is omitted.

Example 1 (*Nonlinear System*). The algorithm developed in this work is implemented on an unknown nonlinear system and compared with the work of Al-Tamimi et al. (2008) using knowledge of the system dynamics. Using the quadratic cost function (2) with $Q(x)=x^{\mathrm{T}}x$ and $R=1$, the nonlinear system

$$x(k+1)=\begin{bmatrix}x_1(k+1)\\x_2(k+1)\end{bmatrix}$$
$$=\begin{bmatrix}-\sin(0.5x_2(k))\\-\cos(1.4x_2(k))\sin(0.9x_1(k))\end{bmatrix}+\begin{bmatrix}0\\1\end{bmatrix}u(k)$$

is considered in simulations. To begin, the identification scheme of Section 3.2 is implemented to learn the nonlinear system dynamics online where ten hidden layer neurons are utilized with the hyperbolic tangent activation functions. The NN learning rate is taken as $\alpha_s=0.09$ while the robustifying adaptation rate was selected as $\gamma_s=0.0027$. Fig. 2 illustrates the results of the online learning phase, and it is observed that the NN identifier successfully learns the nonlinear system dynamics as Theorem 1 predicted.

Now using only the learned system dynamics, the offline HDP algorithm is implemented. The training set is selected from the set defined by $x_1\in[-0.5,0.5]$ and $x_2\in[-0.5,0.5]$ while the action network learning rate was selected to be $\alpha_j=0.01$. Additionally, a sixth order polynomial activation function was used in the value function network. Fig. 3 illustrates the evolution of NN weights for the action and critic networks, respectively, while Fig. 4 shows the NN control input at several different iterations. Examining Fig. 3, convergence of the NN weights is observed, and Fig. 4 shows that at each iteration, the control input is adapting toward its optimal value. The HDP training algorithm converged after 11 iterations.

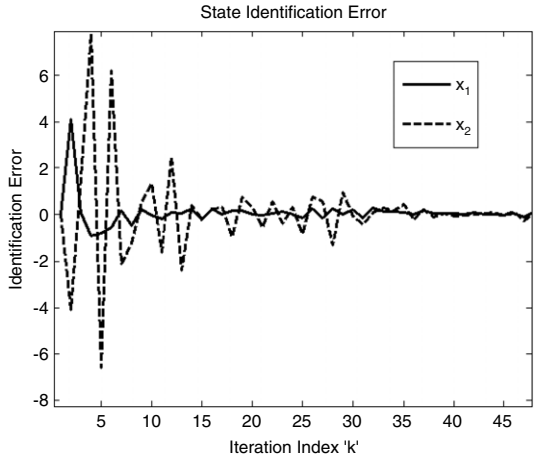

Fig. 2. State identification error for the nonlinear system.

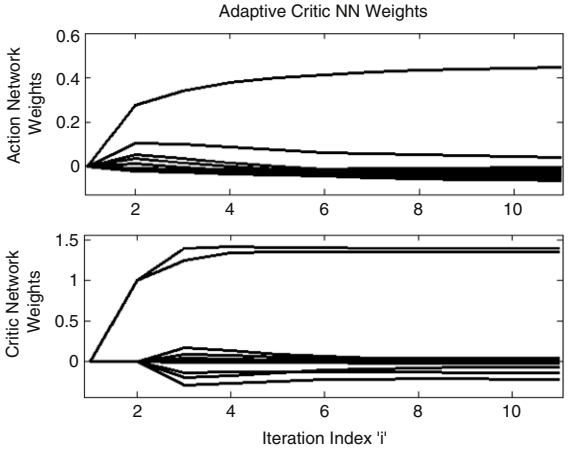

Fig. 3. Actor and critic NN weights.

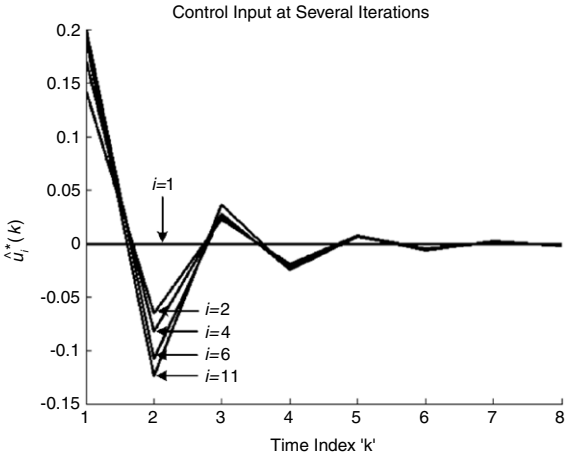

Fig. 4. Optimal controls at various iterations.

Next, the HDP algorithm described in Al-Tamimi et al. (2008) was implemented using full knowledge of the nonlinear system dynamics. Fig. 5 shows the difference between the resulting optimal control laws. In the plots, u^* refers to the control resulting from Al-Tamimi et al. (2008) while $\hat{u}^*$ refers to the control policy resulting from the algorithm derived in this work. Examining Fig. 5, the difference between the two control policies is observed to be of the order of 10^{-3}. Fig. 6 shows the state trajectories resulting from u^* and $\hat{u}^*$, respectively. Examining the results, it is observed there is a very small deviation between the two trajectories due to a result of the uncertainty in the nonlinear system and the estimate of $g(x)$ in (16). Collectively, Figs. 5 and 6 demonstrate that the HDP algorithm can be effectively implemented on unknown nonlinear systems by utilizing a NN identifier to learn the complete system dynamics.

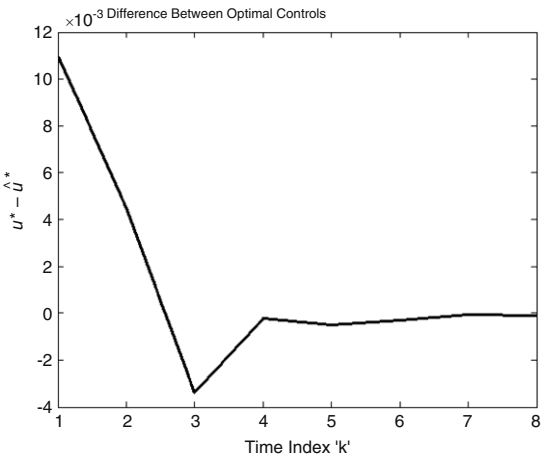

Fig. 5. Difference between optimal controls.

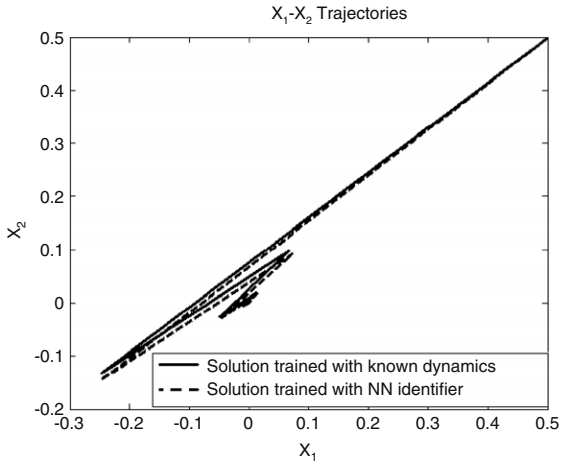

Fig. 6. State trajectories for the nonlinear system.

7. Conclusion

In this work, the optimal control of unknown nonlinear discrete-time systems was undertaken using a two part process: online system identification and offline optimal control training. An asymptotic NN identification scheme was proposed to learn the complete plant dynamics online. Then, using only the NN system model, offline ADP was attempted resulting in a novel optimal control law. The proposed scheme does not require explicit knowledge of the system dynamics as only the learned NN model is needed. Proof of convergence for the NN implementation of the action network was provided, and simulation results were presented to verify theoretical conjecture.

References

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

Chen, Z., & Jagannathan, S. (2008). Generalized Hamilton-Jacobi-Bellman formulation based neural network control of affine nonlinear discrete-time systems. *IEEE Transactions on Neural Networks*, *10*, 90–106.

Dierks, T., Thumati, B. T., & Jagannathan, S. (2009). Adaptive dynamic programming-based optimal control of unknown affine nonlinear discrete-time systems. In *Proceedings of the IEEE international joint conference on neural networks*.

Hayakawa, T., Haddad, W. M., & Hovakimyan, N. (2008). Neural network adaptive control for a class of nonlinear uncertain dynamical systems with asymptotic stability guarantees. *IEEE Transactions on Neural Networks*, *19*, 80–89.

Jagannathan, S. (2006). *Neural network control of nonlinear discrete-time systems*. CRC Press.

Lewis, F. L., & Syrmos, V. L. (1995). *Optimal control* (2nd ed.). Hoboken, NJ: Wiley.

Murray, J. J., Cox, C., Lendaris, G., & Saeks, R. (2002). Adaptive dynamic programming. *IEEE Transactions on Systems, Man and Cybernetics—Part C*, *32*, 140–153.

Prokhorov, D., & Wunsch, D. (1997). Adaptive critic designs. *IEEE Transactions on Neural Networks*, *8*, 997–1007.

Si, J., Barto, A. G., Powell, W. B., & Wunsch, D. (Eds.). (2004). *Handbook of learning and approximate dynamics programming*. Wiley-IEEE Press.