Discrete-Time Nonlinear HJB Solution Using Approximate Dynamic Programming: Convergence Proof

Asma Al-Tamimi, *Student Member, IEEE*, Frank L. Lewis, *Fellow, IEEE*, and Murad Abu-Khalaf, *Member, IEEE*

***Abstract*—Convergence of the value-iteration-based heuristic dynamic programming (HDP) algorithm is proven in the case of general nonlinear systems. That is, it is shown that HDP converges to the optimal control and the optimal value function that solves the Hamilton–Jacobi–Bellman equation appearing in infinite-horizon discrete-time (DT) nonlinear optimal control. It is assumed that, at each iteration, the value and action update equations can be exactly solved. The following two standard neural networks (NN) are used: a critic NN is used to approximate the value function, whereas an action network is used to approximate the optimal control policy. It is stressed that this approach allows the implementation of HDP without knowing the internal dynamics of the system. The exact solution assumption holds for some classes of nonlinear systems and, specifically, in the specific case of the DT linear quadratic regulator (LQR), where the action is linear and the value quadratic in the states and NNs have zero approximation error. It is stressed that, for the LQR, HDP may be implemented without knowing the system A matrix by using two NNs. This fact is not generally appreciated in the folklore of HDP for the DT LQR, where only one critic NN is generally used.**

***Index Terms*—Adaptive critics, approximate dynamic programming (ADP), Hamilton Jacobi Bellman (HJB), policy iteration, value iteration.**

I. Introduction

THIS paper is concerned with the application of approximate dynamic programming (ADP) techniques to solve for the value function, and hence the optimal control policy, in discrete-time (DT) nonlinear optimal control problems having continuous state and action spaces. ADP is a reinforcement learning approach [33] based on adaptive critics [5], [38] to solve dynamic programming problems utilizing function approximation for the value function. ADP techniques can be based on value iterations or policy iterations. In contrast with value iterations, policy iterations require an initial stabilizing control action [33]. Howard [16] proved convergence of policy iteration for Markov decision processes (MDPs) with discrete state and action spaces. Lookup tables are used to store the value function iterations at each state. Watkins [34] developed Q-learning for discrete state and action MDPs, where a "Q function" is stored for each state/action pair, and model dynamics are not needed to compute the control action.

ADP was proposed by Werbos [35]–[37] for DT dynamical systems having continuous state and action spaces as a way to solve optimal control problems [21] forward in time. Bertsekas and Tsitsiklis [6] provide a treatment of neurodynamic programming, where neural networks (NNs) are used to approximate the value function. Cao [39] presents a general theory for learning and optimization.

Werbos [36] classified ADP approaches into the following four main schemes: heuristic dynamic programming (HDP), dual HDP (DHP), action-dependent HDP (ADHDP) (a continuous-state-space generalization of Q-learning [34]), and action-dependent dual HDP (ADDHP). NNs are used to approximate the value function (the critic NN) and the control (the action NN), and backpropagation is used to tune the weights until convergence at each iteration of the ADP algorithm. An overview of ADP is given by Si *et al.* [31] (e.g., [10]) and also Prokhorov and Wunsch [28], who deployed new ADP schemes known as globalized DHP (GDHP) and action-dependent GDHP.

ADP for linear systems has received ample attention. An offline policy iteration scheme for DT systems with known dynamics was given in [14] to solve the DT Riccati equation. In [7], Bradtke *et al.* implemented an (online) Q-learning policy iteration method for DT linear quadratic regulator (LQR) optimal control problems. A convergence proof was given. Hagen and Krose [12] discussed, for the LQR case, the relation between the Q-learning method and model-based adaptive control with system identification. Landelius [20] applied HDP, DHP, ADHDP, and ADDHP value iteration techniques, called greedy policy iterations therein, to the DT LQR problem and verified their convergence. It was shown that these iterations are, in fact, equivalent to iterative solution of an underlying algebraic Riccati equation, which is known to converge (e.g., [19]). Liu and Balakrishnan [24] showed convergence of DHP for the LQR case.

Morimoto *et al.* [25] developed differential dynamic programming, a Q-learning method, to solve optimal zero-sum game problems for nonlinear systems by taking the second-order approximation to the Q-function. This effectively provides an exact Q-learning formulation for linear systems with minimax value functions. In our previous work [3], [4], we studied ADP value iteration techniques to solve the zero-sum

Manuscript received July 18, 2007; revised February 20, 2008. This work was supported in part by the National Science Foundation under Grants ECS-0501451 and ECCS-0801330 and in part by the Army Research Office under Grant W91NF-05-1-0314. This paper was recommended by Guest Editor D. Liu.

A. Al-Tamimi is with The Hashemite University, Zarqa 13115, Jordan (e-mail: altamimi@hu.edu.jo).

F. L. Lewis is with the Automation and Robotics Research Institute, The University of Texas at Arlington, Fort Worth, TX 76118 USA (e-mail: lewis@uta.edu).

M. Abu-Khalaf is with The MathWorks, Inc., Natick, MA 01760 USA (e-mail: Murad.Abu-Khalaf@mathworks.com).

Digital Object Identifier 10.1109/TSMCB.2008.926614

game problem for linear DT dynamical systems using quadratic minimax cost. HDP, DHP, ADHDP, and ADDHP formulations were developed for zero-sum games, and convergence was proven by showing the equivalence of these ADP methods to iterative solution of an underlying game algebraic Riccati equation, which is known to converge. Applications were made to H_∞ control.

For nonlinear systems with continuous state and action spaces, solution methods for the dynamic programming problem are more sparse. Policy iteration methods for optimal control for continuous-time systems with continuous state and action spaces were given in [1] and [2], but complete knowledge of the plant dynamics is required. The DT nonlinear optimal control solution relies on solving the DT Hamilton–Jacobi–Bellman (HJB) equation [21], exact solution of which is generally impossible for nonlinear systems. Solutions to the DT HJB equation with known dynamics and continuous state and action spaces were given in [17], where the coefficients of the Taylor series expansion of the value function are systematically computed. Chen and Jagannathan [8] show that, under certain conditions, a second-order approximation of the DT HJB equation can be considered; under those conditions discussed in that paper, the authors solve for the value function that satisfies the second-order expansion of the DT HJB instead of solving for the original DT HJB. The authors apply a policy iteration scheme on this second-order DT HJB and require an initially stable policy to start the iteration scheme. The authors also used a single (critic) NN to approximate the value function of the second-order DT HJB. These are all offline methods for solving the HJB equations that require full knowledge of the system dynamics.

Convergence proofs for the online value-iteration-based ADP techniques for nonlinear DT systems are even more limited. Prokhorov and Wunsch [28] use NN to approximate both the value (e.g., a critic NN) and the control action. Least mean squares (LMS) is used to tune the critic NN weights and the action NN weights. Stochastic approximation is used to show that, at each iteration of the ADP algorithm, the critic weights converge. Likewise, at each iteration, the action NN weights converge, but the overall convergence of the ADP algorithm to the optimal solution is not demonstrated. A similar approach was used in [30].

In [13], a generalized or asynchronous version of ADP (in the sense of Sutton and Barto [33]) was used, whereby the updates of the critic NN and action NN are interleaved, with each NN being updated at each time step. Tuning was performed online. A Lyapunov approach was used to show that the method yields uniform ultimate bounded stability and that the weight estimation errors are bounded, although convergence to the exact optimal value and control was not shown. The input coupling function must be positive definite.

In this paper, we provide a full rigorous proof of convergence of the value-iteration-based HDP algorithm to solve the DT HJB equation of the optimal control problem for general nonlinear DT systems. It is assumed that, at each iteration, the value update and policy update equations can be exactly solved. Note that this is true in the specific case of the LQR, where the action is linear and the value quadratic in the states. For implementation, two NNs are used—the critic NN to approximate the value and the action NN to approximate the control. Full knowledge of the system dynamics is not needed to implement the HDP algorithm; in fact, the internal dynamics information is not needed. As a value-iteration-based algorithm, of course, an initial stabilizing policy is not needed for HDP.

The point is stressed that these results also hold for the special LQR case of linear systems $\dot{x} = Ax + Bu$ and quadratic utility. In the general folklore of HDP for the LQR case, only a single NN is used, namely, a critic NN, and the action is updated by using a standard matrix equation derived from the stationarity condition [21]. In the DT case, this equation requires the use of both the plant matrix A, e.g., the internal dynamics, and the control-input coupling matrix B. However, by using a second action NN, the knowledge of the A matrix is not needed. This important issue is clarified herein.

Section II starts by introducing the nonlinear DT optimal control problem. Section III demonstrates how to set up the HDP algorithm to solve for the nonlinear DT optimal control problem. In Section IV, we prove the convergence of HDP value iterations to the solution of the DT HJB equation. In Section V, we introduce two NN parametric structures to approximate the optimal value function and policy. As it is known, this provides a procedure for implementing the HDP algorithm. We also discuss in that section how we implement the algorithm without having to know the plant internal dynamics.

II. DT HJB Equation

Consider an affine in input nonlinear dynamical system of the form

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

where $x \in \mathbb{R}^n$, $f(x) \in \mathbb{R}^n$, $g(x) \in \mathbb{R}^{n\times m}$, and the input $u \in \mathbb{R}^m$. Suppose that the system is drift free and, without loss of generality, that $x = 0$ is an equilibrium state, e.g., $f(0) = 0$, and $g(0) = 0$. Assume that the system (1) is stabilizable on a prescribed compact set $\Omega \in \mathbb{R}^n$.

Definition 1: Stabilizable System: A nonlinear dynamical system is defined to be stabilizable on a compact set $\Omega \in \mathbb{R}^n$ if there exists a control input $u \in \mathbb{R}^m$ such that, for all initial conditions $x_0 \in \Omega$, the state $x_k \to 0$ as $k \to \infty$.

It is desired to find the control action $u(x_k)$ which minimizes the infinite-horizon cost function given as

$$V(x_k) = \sum_{n=k}^{\infty} Q(x_n) + u^T(x_n)Ru(x_n) \tag{2}$$

for all x_k, where $Q(x) > 0$ and $R > 0 \in \mathbb{R}^{m\times m}$. The class of controllers needs to be stable and also to guarantee that (2) is finite, i.e., the control must be admissible [1].

Definition 2: Admissible Control: A control $u(x_k)$ is defined to be admissible with respect to (2) on Ω if $u(x_k)$ is continuous on a compact set $\Omega \in \mathbb{R}^n$, $u(0) = 0$, u stabilizes (1) on Ω, and $\forall x_0 \in \Omega$, $V(x_0)$ is finite.

Equation (2) can be written as

$$\begin{aligned} V(x_k) &= x_k^T Q x_k + u_k^T R u_k + \sum_{n=k+1}^{\infty} x_n^T Q x_n + u_n^T R u_n \\ &= x_k^T Q x_k + u_k^T R u_k + V(x_{k+1}) \end{aligned} \tag{3}$$

where we require the boundary condition $V(x = 0) = 0$ so that $V(x_k)$ serves as a Lyapunov function. From Bellman's

optimality principle [21], it is known that, for the infinite-horizon optimization case, the value function $V^*(x_k)$ is time invariant and satisfies the DT HJB equation

$$V^*(x_k) = \min_{u_k} \left(x_k^T Q x_k + u_k^T R u_k + V^*(x_{k+1}) \right). \quad (4)$$

Note that the DT HJB equation develops backward in time.

The optimal control u^* satisfies the first-order necessary condition, which is given by the gradient of the right-hand side of (4) with respect to u as

$$\frac{\partial \left(x_k^T Q x_k + u_k^T R u_k \right)}{\partial u_k} + \frac{\partial x_{k+1}}{\partial u_k}^T \frac{\partial V^*(x_{k+1})}{\partial x_{k+1}} = 0 \quad (5)$$

and therefore

$$u^*(x_k) = \frac{1}{2} R^{-1} g(x_k)^T \frac{\partial V^*(x_{k+1})}{\partial x_{k+1}}. \quad (6)$$

By substituting (6) in (4), the DT HJB becomes

$$V^*(x_k) = x_k^T Q x_k + \frac{1}{4} \frac{\partial V^{*T}(x_{k+1})}{\partial x_{k+1}} g(x_k) R^{-1} g(x_k)^T \times \frac{\partial V^*(x_{k+1})}{\partial x_{k+1}} + V^*(x_{k+1}) \quad (7)$$

where $V^*(x_k)$ is the value function corresponding to the optimal control policy $u^*(x_k)$. This equation reduces to the Riccati equation in the LQR case, which can be efficiently solved. In the general nonlinear case, the HJB cannot be solved exactly.

In the next sections, we apply the HDP algorithm to solve for the value function V^* of the HJB equation (7) and present a convergence proof.

III. HDP Algorithm

The HDP value iteration algorithm [35] is a method to solve the DT HJB online. In this section, a proof of convergence of the HDP algorithm in the general nonlinear DT setting is presented.

A. HDP Algorithm

In the HDP algorithm, one starts with an initial value, e.g., $V_0(x) = 0$, and then solves for u_0 as follows:

$$u_o(x_k) = \arg\min_u \left(x_k^T Q x_k + u^T R u + V_0(x_{k+1}) \right). \quad (8)$$

Once the policy u_0 is determined, iteration on the value is performed by computing

$$\begin{aligned} V_1(x_k) &= x_k^T Q x_k + u_0^T(x_k) R u_0(x_k) + V_0(f(x_k) + g(x_k) u_0(x_k)) \\ &= x_k^T Q x_k + u_0^T(x_k) R u_0(x_k) + V_0(x_{k+1}). \end{aligned} \quad (9)$$

The HDP value iteration scheme, therefore, is a form of incremental optimization that requires iterating between a sequence of action policies $u_i(x)$ determined by the greedy update

$$\begin{aligned} u_i(x_k) &= \arg\min_u \left(x_k^T Q x_k + u^T R u + V_i(x_{k+1}) \right) \\ &= \arg\min_u \left(x_k^T Q x_k + u^T R u + V_i\left(f(x_k) + g(x_k) u \right) \right) \\ &= \frac{1}{2} R^{-1} g(x_k)^T \frac{\partial V_i(x_{k+1})}{\partial x_{k+1}} \end{aligned} \quad (10)$$

and a sequence $V_i(x) \geq 0$ where

$$\begin{aligned} V_{i+1}(x_k) &= \min_u \left(x_k^T Q x_k + u^T R u + V_i(x_{k+1}) \right) \\ &= x_k^T Q x_k + u_i^T(x_k) R u_i(x_k) + V_i(f(x_k) + g(x_k) u_i(x_k)) \end{aligned} \quad (11)$$

with initial condition $V_0(x_k) = 0$.

Note that, as a value iteration algorithm, HDP does not require an initial stabilizing gain. This is important, as stabilizing gains are difficult to find for general nonlinear systems.

Note that i is the value iterations index, whereas k is the time index. The HDP algorithm results in an incremental optimization that is implemented forward in time and online. Note that, unlike the case for policy iterations in [14], the sequence $V_i(x_k)$ is not a sequence of cost functions and is therefore not a sequence of Lyapunov functions for the corresponding policies $u_i(x_k)$ which are, in turn, not necessarily stabilizing. In Section IV, it is shown that $V_i(x_k)$ and $u_i(x_k)$ converge to the value function of the optimal control problem and to the corresponding optimal control policy, respectively.

B. Special Case of Linear Systems

Note that, for the special case of linear systems, it can be shown that the HDP algorithm is one way to solve the DT algebraic Riccati equation (DARE) (e.g., [20]). Particularly, for the DT linear system

$$x_{k+1} = A x_k + B u_k \quad (12)$$

the DT HJB equation (7) becomes the DARE

$$P = A^T P A + Q - A^T P B (R + B^T P B)^{-1} B^T P A \quad (13)$$

with $V^*(x_k) = x_k^T P x_k$.

In the linear case, the policy update (10) is

$$u_i(x_k) = -\left(R + B^T P_i B \right)^{-1} B^T P_i A x_k. \quad (14)$$

By substituting this into (11), one sees that the HDP algorithm (10), (11) is equivalent to

$$\begin{aligned} P_{i+1} &= A^T P_i A + Q - A^T P_i B \left(R + B^T P_i B \right)^{-1} B^T P_i A \\ P_0 &= 0. \end{aligned} \quad (15)$$

It should be noted that the HDP algorithm (15) solves the DARE forward in time, whereas the dynamic programming recursion appearing in finite-horizon optimal control [21] develops backward in time

$$\begin{aligned} P_k &= A^T P_{k+1} A + Q - A^T P_{k+1} B \left(R + B^T P_{k+1} B \right)^{-1} B^T P_{k+1} A \\ P_N &= 0 \end{aligned} \quad (16)$$

where N represents the terminal time. Both (15) and (16) will produce the same sequence of P_i and P_k, respectively. It has been shown in [19] and [21] that this sequence converges to the solution of the DARE after enough iterations.

It is very important to point out the difference between (14) and (15) resulting from HDP value iterations and the following iterations:

$$u_i(x_k) = \underbrace{-\left(R + B^T P_i B \right)^{-1} B^T P_i A}_{K_i} x_k \quad (17)$$

$$(A + B K_i)^T P_{i+1} (A + B K_i) - P_{i+1} = -Q - K_i^T R K_i \quad (18)$$

resulting from policy iterations as those in [14] initialized by a stable K_0. Unlike P_i in (15), the sequence P_i in (18) corresponds to a sequence of Lyapunov functions. Similarly, the sequence of control policies in (17) is stabilizing unlike the sequence in (14).

IV. Convergence of the HDP Algorithm

In this section, we present a proof of convergence for nonlinear HDP. That is, we prove convergence of iterations (10) and (11) to the optimal value, i.e., $V_i \to V^*$ and $u_i \to u^*$, as $i \to \infty$. The linear quadratic case has been proven by Lancaster and Rodman [19] for the case of known system dynamics.

Lemma 1: Let μ_i be any arbitrary sequence of control policies, and let Λ_i be defined by

$$\Lambda_{i+1}(x_k) = Q(x_k) + \mu_i^T R \mu_i + \Lambda_i(\underbrace{f(x_k) + g(x_k)\mu_i(x_k)}_{x_{k+1}}). \tag{19}$$

Let u_i and V_i be the sequences defined by (10) and (11), respectively. If $V_0(x_k) = \Lambda_0(x_k) = 0$, then $V_i(x_k) \le \Lambda_i(x_k)\ \forall i$.

Proof: Because $u_i(x_k)$ minimizes the right-hand side of (11) with respect to the control u, and because $V_0(x_k) = \Lambda_0(x_k) = 0$, then by induction it follows that $V_i(x_k) \le \Lambda_i(x_k)\ \forall i$. ■

Lemma 2: Let the sequence V_i be defined as in (11). If the system is controllable, then the following conditions hold.

1) There exists an upper bound $Y(x_k)$ such that $0 \le V_i(x_k) \le Y(x_k)\ \forall i$.
2) If the optimal control problem (4) is solvable, then there exists a least upper bound $V^*(x_k) \le Y(x_k)$, where $V^*(x_k)$ solves (7) and that $\forall i : 0 \le V_i(x_k) \le V^*(x_k) \le Y(x_k)$.

Proof: Let $\eta(x_k)$ be any stabilizing and admissible control policy, and let $V_0(x_k) = Z_0(x_k) = 0$, where Z_i is updated as

$$\begin{aligned} Z_{i+1}(x_k) &= Q(x_k) + \eta^T(x_k) R \eta(x_k) + Z_i(x_{k+1}) \\ x_{k+1} &= f(x_k) + g(x_k)\eta(x_k). \end{aligned} \tag{20}$$

It follows that the difference

$$\begin{aligned} Z_{i+1}(x_k) - Z_i(x_k) &= Z_i(x_{k+1}) - Z_{i-1}(x_{k+1}) \\ &= Z_{i-1}(x_{k+2}) - Z_{i-2}(x_{k+2}) \\ &= Z_{i-2}(x_{k+3}) - Z_{i-3}(x_{k+3}) \\ &\ \ \vdots \\ &= Z_1(x_{k+i}) - Z_0(x_{k+i}). \end{aligned} \tag{21}$$

Because $Z_0(x_k) = 0$, it then follows that

$$\begin{aligned} Z_{i+1}(x_k) &= Z_1(x_{k+i}) + Z_i(x_k) \\ &= Z_1(x_{k+i}) + Z_1(x_{k+i-1}) + Z_{i-1}(x_k) \\ &= Z_1(x_{k+i}) + Z_1(x_{k+i-1}) + Z_1(x_{k+i-1}) + Z_{i-2}(x_k) \\ &= Z_1(x_{k+i}) + Z_1(x_{k+i-1}) \\ &\quad + Z_1(x_{k+i-2}) + \cdots + Z_1(x_k) \end{aligned} \tag{22}$$

and (22) can be written as

$$\begin{aligned} Z_{i+1}(x_k) &= \sum_{n=0}^{i} Z_1(x_{k+n}) \\ &= \sum_{n=0}^{i} \left(Q(x_{k+n}) + \eta^T(x_{k+n}) R \eta(x_{k+n})\right) \\ &\le \sum_{n=0}^{\infty} \left(Q(x_{k+n}) + \eta^T(x_{k+n}) R \eta(x_{k+n})\right). \end{aligned} \tag{23}$$

Because $\eta(x_k)$ is an admissible stabilizing controller, $x_{k+n} \to 0$, as $n \to \infty$, and

$$\forall i : \quad Z_{i+1}(x_k) \le \sum_{i=0}^{\infty} Z_1(x_{k+i}) = Y(x_k).$$

By using Lemma 1 with $\mu_i(x_k) = \eta(x_k)$ and $\Lambda_i(x_k) = Z_i(x_k)$, it follows that $\forall i : V_i(x_k) \le Z_i(x_k) \le Y(x_k)$, which proves part 1). Moreover, if $\eta(x_k) = u^*(x_k)$, then

$$\underbrace{\sum_{n=0}^{\infty} \left(Q(x_{k+n}) + u^{*T}(x_{k+n}) R u^*(x_{k+n})\right)}_{V^*(x_k)} \le \underbrace{\sum_{n=0}^{\infty} \left(Q(x_{k+n}) + \eta^T(x_{k+n}) R \eta(x_{k+n})\right)}_{Y(x_k)}$$

and hence, $V^*(x_k) \le Y(x_k)$, which proves part 1) and shows that $\forall i : 0 \le V_i(x_k) \le V^*(x_k) \le Y(x_k)$ for any $Y(x_k)$ determined by an admissible stabilizing policy $\eta(x_k)$. ■

We now present our main result.

Theorem 1: Consider sequences V_i and u_i defined by (10) and (11), respectively. If $V_0(x_k) = 0$, then it follows that V_i is a nondecreasing sequence $\forall i : V_{i+1}(x_k) \ge V_i(x_k)$. Moreover, as $i \to \infty$, $V_i \to V^*$, $u_i \to u^*$, and hence, the sequence V_i converges to the solution of the DT HJB (7).

Proof: From Lemma 1, let μ_i be any arbitrary sequence of control policies, and let Λ_i be defined by

$$\Lambda_{i+1}(x_k) = Q(x_k) + \mu_i^T R \mu_i + \Lambda_i(\underbrace{f(x_k) + g(x_k)\mu_i(x_k)}_{x_{k+1}}).$$

If $V_0(x_k) = \Lambda_0(x_k) = 0$, then it follows that $V_i(x_k) \le \Lambda_i(x_k)\ \forall i$. Now, assume that $\mu_i(x_k) = u_{i+1}(x_k)$ such that

$$\begin{aligned} \Lambda_{i+1}(x_k) &= Q(x_k) + \mu_i^T R \mu_i + \Lambda_i(f(x_k) + g(x_k)\mu_i(x_k)) \\ &= Q(x_k) + u_{i+1}^T R u_{i+1} + \Lambda_i(f(x_k) + g(x_k)u_{i+1}(x_k)) \end{aligned} \tag{24}$$

and consider

$$V_{i+1}(x_k) = Q(x_k) + u_i^T R u_i + V_i(f(x_k) + g(x_k)u_i(x_k)). \tag{25}$$

It will next be proven by induction that if $V_0(x_k) = \Lambda_0(x_k) = 0$, then $\Lambda_i(x_k) \le V_{i+1}(x_k)$. Induction is initialized by letting $V_0(x_k) = \Lambda_0(x_k) = 0$ and hence

$$\begin{aligned} V_1(x_k) - \Lambda_0(x_k) &= Q(x_k) \ge 0 \\ V_1(x_k) &\ge \Lambda_0(x_k). \end{aligned}$$

Now, assume that $V_i(x_k) \geq \Lambda_{i-1}(x_k)$. Then, by subtracting (24) from (25), it follows that

$$V_{i+1}(x_k) - \Lambda_i(x_k) = V_i(x_{k+1}) - \Lambda_{i-1}(x_{k+1}) \geq 0$$

and this completes the proof that $\Lambda_i(x_k) \leq V_{i+1}(x_k)$.

From $\Lambda_i(x_k) \leq V_{i+1}(x_k)$ and $V_i(x_k) \leq \Lambda_i(x_k)$, it then follows that $\forall i : V_i(x_k) \leq V_{i+1}(x_k)$.

From part 1) in Lemma 2 and the fact that V_i is a nondecreasing sequence, it follows that $V_i \to V_\infty$, as $i \to \infty$. From part 2) of Lemma 2, it also follows that $V_\infty(x_k) \leq V^*(x_k)$.

It now remains to show that, in fact, V_∞ is V^*. To see this, note that, from (11), it follows that

$$V_\infty(x_k) = x_k^T Q x_k + u_\infty^T(x_k) R u_\infty(x_k) + V_\infty\left(f(x_k) + g(x_k)u_\infty(x_k)\right)$$

and hence

$$V_\infty\left(f(x_k) + g(x_k)u_\infty(x_k)\right) - V_\infty(x_k) = -x_k^T Q x_k - u_\infty^T(x_k) R u_\infty(x_k)$$

and therefore, $V_\infty(x_k)$ is a Lyapunov function for a stabilizing and admissible policy $u_\infty(x_k) = \eta(x_k)$. By using part 2) of Lemma 2, it follows that $V_\infty(x_k) = Y(x_k) \geq V^*(x_k)$. This implies that $V^*(x_k) \leq V_\infty(x_k) \leq V^*(x_k)$ and, hence, $V_\infty(x_k) = V^*(x_k)$, $u_\infty(x_k) = u^*(x_k)$. ■

V. NN Approximation for Value and Action

We have just proven that the nonlinear HDP algorithm converges to the value function of the DT HJB equation that appears in the nonlinear DT optimal control. It was assumed that the action and value update equations (10) and (11) can be exactly solved at each iteration. In fact, these equations are difficult to solve for general nonlinear systems. Therefore, for implementation purposes, one needs to approximate u_i and V_i at each iteration. This allows the approximate solutions of (10) and (11).

In this section, we review how to implement the HDP value iteration algorithm with two parametric structures such as NNs [22], [37]. The important point is stressed that the use of two NNs, a critic for value function approximation and an action NN for the control, allows the implementation of HDP in the LQR case without knowing the system internal dynamics matrix A.

A. NN Approximation for Implementation of the HDP Algorithm for Nonlinear Systems

It is well known that NNs can be used to approximate smooth functions on prescribed compact sets [15]. Therefore, to solve (10) and (11), $V_i(x)$ is approximated at each step by a critic NN

$$\hat{V}_i(x) = \sum_{j=1}^{L} w_{vi}^j \phi_j(x) = W_{Vi}^T \phi(x) \tag{26}$$

and $u_i(x)$ by an action NN

$$\hat{u}_i(x) = \sum_{j=1}^{M} w_{ui}^j \sigma_j(x) = W_{ui}^T \sigma(x) \tag{27}$$

where the activation functions are $\phi_j(x)$, $\sigma_j(x) \in C^1(\Omega)$, respectively. Because it is required that $V_i(0) = 0$ and $u_i(0) = 0$, we select activation functions with $\phi_j(0) = 0$ and $\sigma_j(0) = 0$. Moreover, because it is known that V^* is a Lyapunov function and that Lyapunov proofs are convenient if the Lyapunov function is symmetric and positive definite, it is convenient to also require that the activation functions for the critic NN be symmetric, i.e., $\phi_j(x) = \phi_j(-x)$.

The NN weights in the critic NN (26) are w_{vi}^j. L is the number of hidden-layer neurons. The vector $\phi(x) \equiv [\phi_1(x)\ \phi_2(x)\ \cdots\ \phi_L(x)]^{\mathrm{T}}$ is the vector activation function, and $W_{Vi} \equiv [w_{vi}^1\ w_{vi}^2\ \cdots\ w_{vi}^L]^{\mathrm{T}}$ is the weight vector at iteration i. Similarly, the weights of the NN in (27) are w_{ui}^j. M is the number of hidden-layer neurons. $\boldsymbol{\sigma}(x) \equiv [\sigma_1(x)\ \sigma_2(x)\ \cdots\ \sigma_L(x)]^{\mathrm{T}}$ is the vector activation function, and $W_{ui} \equiv [w_{ui}^1\ w_{ui}^2\ \cdots\ w_{ui}^L]^{\mathrm{T}}$ is the vector weight.

According to (11), the critic weights are tuned at each iteration of HDP to minimize the residual error between $\hat{V}_{i+1}(x_k)$ and the target function defined in (28) in a least squares sense for a set of states x_k sampled from a compact set $\Omega \subset \mathbb{R}^n$

$$\begin{aligned} d(x_k, x_{k+1}, W_{Vi}, W_{ui}) &= x_k^T Q x_k + \hat{u}_i^T(x_k) R \hat{u}_i(x_k) + \hat{V}_i(x_{k+1}) \\ &= x_k^T Q x_k + \hat{u}_i^T(x_k) R \hat{u}_i(x_k) \\ &\quad + W_{Vi}^T \boldsymbol{\phi}(x_{k+1}). \end{aligned} \tag{28}$$

The residual error (cf. temporal difference error) becomes

$$\left(W_{Vi+1}^T \boldsymbol{\phi}(x_k) - d(x_k, x_{k+1}, W_{Vi}, W_{ui})\right) = e_L(x). \tag{29}$$

Note that the residual error in (29) is explicit, in fact linear, in the tuning parameters W_{Vi+1}. Therefore, to find the least squares solution, the method of weighted residuals may be used [11]. The weights W_{Vi+1} are determined by projecting the residual error onto $(de_L(x)/dW_{Vi+1})$ and setting the result to zero $\forall x \in \Omega$ using the inner product, i.e.,

$$\left\langle \frac{de_L(x)}{dW_{Vi+1}}, e_L(x) \right\rangle = 0 \tag{30}$$

where $\langle \mathrm{f}, \mathrm{g} \rangle = \int_\Omega f g^T dx$ is a Lebesgue integral. One has

$$0 = \int_\Omega \phi(x_k)\left(\phi^T(x_k) W_{Vi+1} - d^T(x_k, x_{k+1}, W_{Vi}, W_{ui})\right) dx_k. \tag{31}$$

Therefore, a unique solution for W_{Vi+1} exists and is computed as

$$W_{Vi+1} = \left(\int_\Omega \phi(x_k)\phi(x_k)^T dx\right)^{-1} \times \int_\Omega \phi(x_k) d^T\left(\phi(x_k), W_{Vi}, W_{ui}\right) dx. \tag{32}$$

To use this solution, it is required that the outer product integral be positive definite. This is known as a persistence of excitation condition in system theory. The next assumption is standard in selecting the NN activation functions as a basis set.

Assumption 1: The selected activation functions $\{\phi_j(x)\}^L$ are linearly independent on the compact set $\Omega \subset \mathbb{R}^n$.

Assumption 1 guarantees that the excitation condition is satisfied, and hence, $\int_\Omega \phi(x_k)\phi(x_k)^T dx$ is of full rank and invertible, and a unique solution for (32) exists.

The action NN weights are tuned to solve (10) at each iteration. The use of $\hat{u}_i(x_k, W_{ui})$ from (27) allows the rewriting of (10) as

$$W_{ui} = \arg\min_w \left(x_k^T Q x_k + \hat{u}_i^T(x_k, w) R \hat{u}_i(x_k, w) + \hat{V}_i(x_{k+1}^i) \right) \Big|_\Omega \tag{33}$$

where $x_{k+1}^i = f(x_k) + g(x_k)\hat{u}_i(x_k, w)$, and the notation means minimization for a set of points x_k selected from the compact set $\Omega \in \mathbb{R}^n$.

Note that the control weights W_{ui} appear in (33) in an implicit fashion, i.e., it is difficult to solve explicitly for the weights because the current control weights determine x_{k+1}. Therefore, one can use an LMS algorithm on a training set constructed from Ω. The weight update is therefore (34), shown at the bottom of the page, where α is a positive step size and m is the iteration number for the LMS algorithm. By a stochastic-approximation-type argument, the weights $W_{ui}|_m \Rightarrow W_{ui}$, as $m \Rightarrow \infty$, and satisfy (33). Note that one can use alternative tuning methods, such as Newton's method and the Levenberg–Marquardt method, in order to solve (33).

In Fig. 1, the flowchart of the HDP iteration is shown. Note that because of the NN used to approximate the control policy, the internal dynamics, i.e., $f(x_k)$, is not needed. That is, the internal dynamics can be unknown.

Remark: Neither $f(x)$ nor $g(x)$ is needed to update the critic NN weights using (32). Only the input coupling term $g(x)$ is needed to update the action NN weights using (34). Therefore, the proposed algorithm works for a system with partially unknown dynamics—no knowledge of the internal feedback structure $f(x)$ is needed.

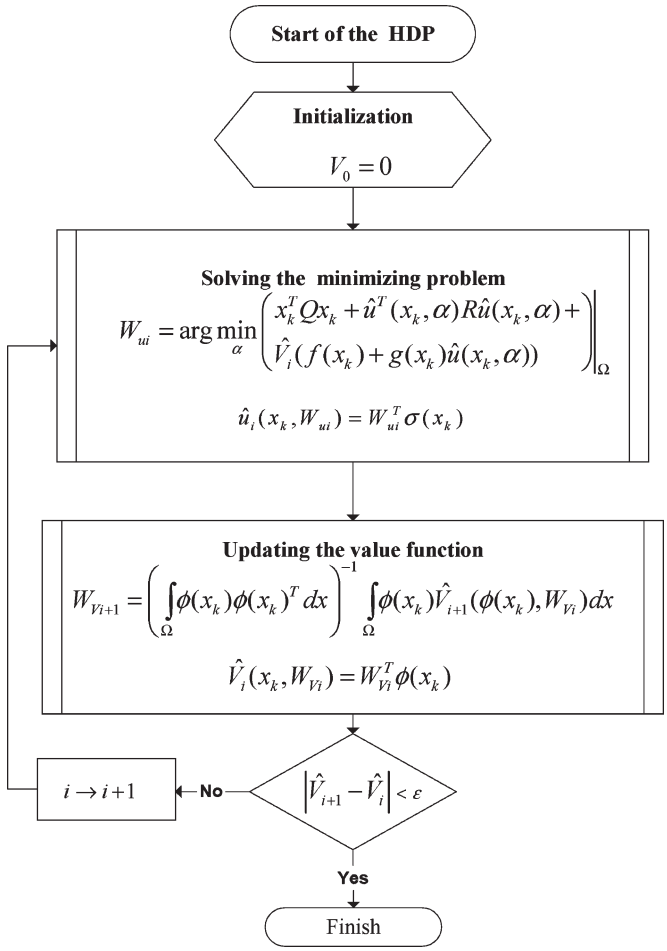

Fig. 1. Flowchart of the proposed algorithm.

B. HDP for Linear Systems Without Knowledge of Internal Dynamics

The general practice in the HDP folklore for linear quadratic systems is to use a critic NN (to approximate the value) and update the critic weights using a method such as the batch update (32) or a recursive update method such as LMS. In fact, the critic weights are nothing, but the elements of the Riccati matrix and the activation functions are quadratic polynomials in terms of the states. Then, the policy is updated by using

$$u_i(x_k) = -\left(R + B^T P_i B\right)^{-1} B^T P_i A x_k. \tag{35}$$

Note that this equation requires the full knowledge of both the internal dynamics matrix A and the control weighting matrix B. However, we have just seen (see the previous remark) that the knowledge of the A matrix can be avoided by using, instead of the action update (35), a second NN for the action $\hat{u}_i(x) = W_{ui}^T \boldsymbol{\sigma}(x)$.

In fact, the action NN approximates the effects of A and B given in (35) and so effectively learns the A matrix.

That is, using two NNs even in the LQR case avoids the need to know the internal dynamics matrix A. Only the input coupling matrix B is needed for the HDP algorithm, which nevertheless converges to the correct LQR Riccati solution matrix P.

VI. Conclusion

We have proven convergence of the HDP algorithm to the value function solution of the HJB equation for nonlinear dynamical systems, assuming exact solution of the value update and the action update at each iteration.

$$W_{ui}|_{m+1} = W_{ui}|_m - \alpha \frac{\partial (x_k^T Q x_k + \hat{u}_i^T(x_k, W_{ui}|_m)\, R\, \hat{u}_i(x_k, W_{ui}|_m) + \hat{V}_i(x_{k+1})}{\partial W_{ui}} \Bigg|_{W_{ui}|_m}$$

$$= W_{ui}|_m - \alpha \sigma(x_k) \left(2R\hat{u}_i(x_k, W_{ui}|_m) + g(x_k)^T \frac{\partial \phi(x_{k+1})}{\partial x_{k+1}} W_{Vi} \right)^T \tag{34}$$

NNs are used as parametric structures to approximate at each iteration the value (i.e., critic NN) and the control action. It is stressed that the use of the second NN to approximate the control policy, the internal dynamics, i.e., $f(x_k)$, is not needed to implement HDP. This holds as well for the special LQR case, where the use of two NNs avoids the need to know the system internal dynamics matrix A.

REFERENCES

[1] 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.

[2] M. Abu-Khalaf, F. L. Lewis, and J. Huang, "Hamilton–Jacobi–Isaacs formulation for constrained input nonlinear systems," in *Proc. 43rd IEEE Conf. Decision Control*, Paradise Island, Bahamas, 2004, vol. 5, pp. 5034–5040.

[3] 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.

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

[5] A. G. Barto, R. S. Sutton, and C. W. Anderson, "Neuron-like adaptive elements that can solve difficult learning control problems," *IEEE Trans. System, Man, Cybern.*, vol. SMC-13, no. 5, pp. 835–846, 1983.

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

[7] S. J. Bradtke, B. E. Ydestie, and A. G. Barto, "Adaptive linear quadratic control using policy iteration," in *Proc. Amer. Control Conf.*, Baltimore, MD, Jun. 1994, pp. 3475–3476.

[8] Z. Chen and S. Jagannathan, "Generalized Hamilton–Jacobi–Bellman formulation-based neural network control of affine nonlinear discrete-time systems," *IEEE Trans. Neural Netw.*, vol. 10, no. 1, pp. 90–106, Jan. 2008.

[9] J. R. Cloutier, "State-dependent Riccati equation techniques: An overview," in *Proc. Amer. Control Conf.*, Albuquerque, NM, Jun. 4–6, 1997, pp. 932–936.

[10] S. Ferrari and R. Stengel, "Model-based adaptive critic designs," in *Handbook of Learning and Approximate Dynamic Programming*, J. Si, A. Barto, W. Powell, and D. Wunsch, Eds. Hoboken, NJ: Wiley, Aug. 2004, pp. 64–94.

[11] B. A. Finlayson, *The Method of Weighted Residuals and Variational Principles*. New York: Academic, 1972.

[12] S. Hagen and B. Krose, "Linear quadratic regulation using reinforcement learning," in *Proc. Belgian Dutch Conf. Mech. Learn.*, 1998, pp. 39–46.

[13] 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.

[14] G. A. Hewer, "An iterative technique for the computation of the steady state gains for the discrete optimal regulator," *IEEE Trans. Autom. Control*, vol. AC-16, no. 4, pp. 382–384, Aug. 1971.

[15] K. Hornik, M. Stinchcombe, and H. White, "Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks," *Neural Netw.*, vol. 3, no. 5, pp. 551–560, 1990.

[16] R. Howard, *Dynamic Programming and Markov Processes*. Cambridge, MA: MIT Press, 1960.

[17] J. Huang, "An algorithm to solve the discrete HJI equation arising in the L_2-gain optimization problem," *Int. J. Control*, vol. 72, no. 1, pp. 49–57, Jan. 1999.

[18] W. H. Kwon and S. Han, *Receding Horizon Control*. London, U.K.: Springer-Verlag, 2005.

[19] P. L. Lancaster and Rodman, *Algebraic Riccati Equations*. London, U.K.: Oxford Univ. Press, 1995.

[20] T. Landelius, "Reinforcement learning and distributed local model synthesis," Ph.D. dissertation, Linkoping Univ., Linkoping, Sweden, 1997.

[21] F. L. Lewis and V. L. Syrmos, *Optimal Control*, 2nd ed. Hoboken, NJ: Wiley, 1995.

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

[23] W. Lin and C. I. Byrnes, "H_∞ control of discrete-time nonlinear systems," *IEEE Trans. Autom. Control*, vol. 41, no. 4, pp. 494–510, Apr. 1996.

[24] X. Lu and S. N. Balakrishnan, "Convergence analysis of adaptive critic based optimal control," in *Proc. Amer. Control Conf.*, Chicago, IL, 2000, pp. 1929–1933.

[25] J. Morimoto, G. Zeglin, and C. G. Atkeson, "Minimax differential dynamic programming: Application to a biped walking robot," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, Las Vegas, NV, 2003, pp. 1927–1932.

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

[27] K. S. Narendra and F. L. Lewis, "Introduction to the special issue on neural network feedback control," *Automatica*, vol. 37, no. 8, pp. 1147–1148, Aug. 2001.

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

[29] D. Prokhorov and D. Wunsch, "Convergence of critic-based training," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, 1997, vol. 4, pp. 3057–3060.

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

[31] S. Ji, A. Barto, W. Powell, and D. Wunsch, *Handbook of Learning and Approximate Dynamic Programming*. Hoboken, NJ: Wiley, 2004.

[32] B. Stevens and F. L. Lewis, *Aircraft Control and Simulation*, 2nd ed. Hoboken, NJ: Wiley, 2003.

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

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

[35] P. J. Werbos, "A menu of designs for reinforcement learning over time," in *Neural Networks for Control*, W. T. Miller, R. S. Sutton, and P. J. Werbos, Eds. Cambridge, MA: MIT Press, 1991, pp. 67–95.

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

[37] P. J. Werbos, "Neural networks for control and system identification," *Heuristics*, vol. 3, no. 1, pp. 18–27, Spring 1990.

[38] B. Widrow, N. Gupta, and S. Maitra, "Punish/reward: Learning with a critic in adaptive threshold systems," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-3, no. 5, pp. 455–465, Sep. 1973.

[39] X.-R. Cao, "Learning and optimization—From a systems theoretic perspective," in *Proc. IEEE Conf. Decision Control*, 2002, pp. 3367–3371.

Author photographs and biographies not available at the time of publication.