Iterative Adaptive Dynamic Programming for Solving Unknown Nonlinear Zero-Sum Game Based on Online Data

Yuanheng Zhu, *Member, IEEE*, Dongbin Zhao, *Senior Member, IEEE*, Xiangjun Li, *Senior Member, IEEE*

***Abstract*—H_∞ control is a powerful method to solve disturbance attenuation problems that occur in some control systems. The design of such controllers relies on solving the zero-sum game. But in practical applications, the exact dynamics is mostly unknown. Identification of dynamics also produces errors that are detrimental to the control performance. To overcome this problem, an iterative adaptive dynamic programming algorithm is proposed in this paper to solve the continuous-time, unknown nonlinear zero-sum game with only online data. A model-free approach to the Hamilton-Jacobi-Isaacs equation is developed based on the policy iteration method. Control and disturbance policies and value are approximated by neural networks under the critic-actor-disturber structure. The neural network weights are solved by the least-squares method. According to theoretical analysis, our algorithm is equivalent to a Gauss-Newton method solving an optimization problem, and it converges uniformly to the optimal solution. The online data can also be used repeatedly, which is highly efficient. Simulation results demonstrate its feasibility to solve the unknown nonlinear zero-sum game. When compared with other algorithms, it saves a significant amount of online measurement time.**

***Index Terms*—H_∞ control, zero-sum game, adaptive dynamic programming, policy iteration**

I. Introduction

OPTIMAL control [1], [2], [3] has always been a hot topic in control theory. Its goal is to find a policy that minimizes a predefined performance index. In various control applications, there are numerous situations where disturbance exists in systems and plays a negative role in the control performance. H_∞ control [4], [5] provides a powerful tool to reduce disturbance effect. According to the game theory [6], finding the H_∞ controller is equivalent to solving a two-player zero-sum game (ZSG) [7] where the controller attempts to minimize the performance index under the worst possible disturbance. For a system with continuous-time (CT) nonlinear dynamics, solution to the ZSG can be obtained by solving the Hamilton-Jacobi-Isaacs (HJI) equation [8]. However, it is almost impossible due to the inherent nonlinearity.

Many approximation methods have been proposed to solve the HJI equation [9], [10]. Recently, a newly developed technique, adaptive/approximate dynamic programming (ADP) [11], [12], [13], [14], [15], [16], [17], [18], [19], has solved various optimal control problems with a focus on the ZSG problem. For example, Abu-Khalaf et al. proposed an offline inner-outer-loop policy iteration (PI) algorithm to solve the constrained HJI equation in [20], [21]. It was proved that policies in the algorithm were convergent to the optimal solution. Zhang et al. [22] studied a specific case in which the saddle point might not exist. Their work resulted in a mixed optimal control pair. Note that both these methods are offline. In [23], Dierks and Jagannathan used a single online approximator to address the ZSG. It was proved that both the states and the approximation errors were uniformly ultimately bounded (UUB). Vamvoudakis and Lewis [24] proposed an online ADP algorithm which extended their synchronous policy iteration algorithm (SPIA) [25] to the ZSG with a critic-actor-disturber neural-network (NN) structure. Unfortunately, these two works still require complete knowledge of system dynamics. Wu and Luo [26], [27] went further to use the idea of integral reinforcement learning (IRL) that removed the dependence on the internal dynamics. They proposed a simultaneous policy update algorithm which only included 1-loop iteration as opposed to 2-loop iteration used in [20], [21]. In [28], Yasini et al. combined IRL with synchronous PI and applied concurrent learning in their concurrent reinforcement learning algorithm (CRLA), which improved the learning speed significantly. For the discrete-time (DT) ZSG, ADP is also successful in [29], [30], [31], [32].

In contrast to classical dynamic programming, ADP solves dynamic programming in a forward-in-time way. ADP is developed from reinforcement learning (RL) [33], [34], [35], [36], and is capable to deal with various control problems. As an example, the famous Hamilton-Jacobi-Bellman (HJB) equation was addressed in [37], [38], [39]. For the CT multi-player non-zero-sum (NZS) game, [40] proposed an online adaptive control solution based on policy iteration. Zhang et al. designed an algorithm to learn the Nash equilibrium of the NZS game with only the critic network [41]. We introduced experience replay to solve the unknown NZS game [42].

In practical applications, the exact mathematical dynamics is usually unknown. Some researchers employed NNs to identify unknown dynamics, and then applied ADP on the identifier

Y. Zhu is with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China (e-mail: yuanheng.zhu@ia.ac.cn). D. Zhao is with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, and with University of Chinese Academy of Sciences (e-mail: dongbin.zhao@ia.ac.cn). X. Li is with State Key Laboratory of Control and Operation of Renewable Energy and Storage Systems, China Electric Power Research Institute, Beijing 100192, China.

This work is supported by National Natural Science Foundation of China (NSFC) under Grants No. 61273136, No. 61573353 and No. 61533017, by Beijing Nova Program under Grant No. Z141101001814094, and by Science and Technology Foundation of SGCC under Grant No. DG71-14-032.

networks to find the optimal solutions [32], [43], [44], [45], [46], [47]. Unfortunately, identification errors in networks are detrimental to the optimality of final controllers. Training identifier networks also increases the computational cost and increases the learning time. A totally model-free approach is more efficient. In [48], [49], Jiang and Jiang proposed a robust ADP to solve the nonlinear optimal control problem without any dynamics. The uniform convergence to the optimal solution was proved in their work. As for linear quadratic ZSG with linear dynamics and quadratic performance index, the HJI equation is reduced to the generalized algebraic Riccati equation (GARE). Vrabie and Lewis [50] used a model-free algorithm to find its solution. Another research study of the same problem was presented in [51]. As for nonlinear ZSG, [52] developed an online off-policy RL algorithm to learn the H_∞ tracking controller for unknown CT systems. The off-policy IRL Bellman equation could iteratively solve the tracking HJI equation. A sequence of control and disturbance policies and values were obtained, and proved to converge to the optimal solutions. When implementing, values, controllers and disturbers were approximated by NNs. Their weights were determined by the least-squares (LS) method. The convergence property of NN approximation was not discussed in their work, which motivated our research.

In this paper, a continuous-time unknown nonlinear zero-sum game is considered and an iterative ADP algorithm is designed to learn the H_∞ controller without system dynamics. Policy iteration and integral reinforcement learning techniques are used to iteratively solve the HJI equation. Three NNs are used to approximate the control and disturbance policies and the value. Their weights are determined by the least-squares method with online data. It is proved that NN weights are convergent uniformly to the optimal solutions. Two examples are used to test the algorithm performance. The measurement time is significantly reduced when compared with other algorithms.

The rest of this paper is organized as follows: Section II briefly introduces the nonlinear ZSG and the iterative approach for solving the HJI equation. Section III describes a model-free algorithm. Section IV is convergence analysis. Section V is a summary of the algorithm implementation. Section VI gives out simulation results to illustrate the effectiveness of this algorithm. Section VII presents final conclusions.

II. Introduction to Nonlinear ZSG and Iterative Approach for Solving the HJI Equation

A. Continuous-time nonlinear zero-sum game

Consider the following continuous-time nonlinear system

$$\begin{aligned} \dot{x} &= f(x) + g(x)u + k(x)\omega \\ z &= h(x) \end{aligned} \tag{1}$$

where $x(t) \in \mathbb{R}^n$ is the state vector, $u(t) \in \mathbb{R}^m$ is the control signal, $\omega(t) \in \mathbb{R}^q$ is the external disturbance satisfying $\omega(t) \in L_2[0,\infty)$, $z \in \mathbb{R}^p$ is the fictitious output, and $f(x)$, $g(x)$, $k(x)$ are the system dynamics vector and matrices with appropriate dimensions. Assume that $f(x)$, $g(x)$, and $k(x)$ are Lipschitz continuous and $f(0) = 0$. Thus, $x = 0$ is the equilibrium of the system.

H_∞ control is to find a controller that renders the performance index

$$J(x(0), u, \omega) = \int_0^\infty \left(h^T(x)h(x) + u^T R u - \gamma^2 \omega^T \omega\right) d\tau$$

nonpositive for all $\omega \in L_2[0,\infty)$ with $x(0) = 0$, where $R > 0$, $\gamma \geq \gamma^* \geq 0$. If such a controller exists, it is said that the system has L_2-gain $\leq \gamma$. γ^* represents the smallest value for which the problem is still solvable.

In this paper, we only concentrate on feedback policies with complete state information. Given a control policy $u(t) \equiv u(x(t))$ and a disturbance policy $\omega(t) \equiv \omega(x(t))$, their value is defined as

$$\begin{aligned} V(x(0)) &= \int_0^\infty \left(h^T h + u^T R u - \gamma^2 \omega^T \omega\right) d\tau \\ &\equiv \int_0^\infty r(x(t), u, \omega) d\tau \end{aligned}$$

A differential equivalent of the above definition is a Lyapunov equation

$$r(x, u, \omega) + \nabla V^T (f + gu + k\omega) = 0, V(0) = 0 \tag{2}$$

where ∇ denotes the partial derivative operator, i.e. $\nabla V = \partial V / \partial x$. Hamiltonian function is defined as

$$H(x, \nabla V, u, \omega) \equiv r(x, u, \omega) + \nabla V^T (f + gu + k\omega)$$

Based on the game theory, finding the H_∞ controller is equivalent to solving a two-player zero-sum game. The control signal is to maximize the performance index while the disturbance signal is to minimize it. The continuous-time nonlinear ZSG is to find the feedback control and disturbance policies with the optimal value

$$V^*(x) \equiv \min_u \max_\omega J(x, u, \omega)$$

If the following Nash condition is satisfied

$$\min_u \max_\omega J(x, u, \omega) = \max_\omega \min_u J(x, u, \omega)$$

ZSG has a unique solution which is termed as the saddle point (u^*, ω^*), and u^* is an H_∞ controller for (1). Furthermore, according to the stationary condition, u^* and ω^* are

$$\begin{aligned} u^* &= -\frac{1}{2} R^{-1} g^T \nabla V^* \\ \omega^* &= \frac{1}{2} \gamma^{-2} k^T \nabla V^* \end{aligned}$$

After substituting u^* and ω^* into the Lyapunov equation (2), we obtain the Hamilton-Jacobi-Isaacs equation

$$\begin{aligned} \nabla V^{*T} f + h^T h - \frac{1}{4} \nabla V^{*T} g R^{-1} g^T \nabla V^* \\ + \frac{1}{4} \gamma^{-2} \nabla V^{*T} k k^T \nabla V^* = 0, V^*(0) = 0 \end{aligned} \tag{3}$$

Assumption 1: [24] Select $\gamma > 0$. System (1) is zero-state observable. There exists a control policy $u(x)$ for which the system has L_2-gain $\leq \gamma$ on a set $\Omega \in \mathbb{R}^n$ and is asymptotically stable. There exists a smooth solution for the HJI equation (3) on Ω.

B. Iterative approach for solving the HJI equation

For nonlinear systems, the HJI equation is a nonlinear partial differential equation (PDE) which has no global analytic solution. Policy iteration is widely used to approximately solve the equation. In [26], Wu and Luo proposed a simultaneous policy update algorithm (SPUA). It only included 1-loop iteration that is computationally efficient compared to the inner-outer-loop policy iteration in [20], [21]. The next paragraph is a simplified SPUA algorithm.

Given an appropriate initial value V_0, a sequence of $\{V_i\}_{i=0}^{\infty}$ is produced by iterating on the following Lyapunov equation with $V_i(0)=0$

$$\nabla V_i^T(f+gu_i+k\omega_i)+h^Th+u_i^TRu_i-\gamma^2\omega_i^T\omega_i=0 \quad (4)$$

and the policy update laws

$$u_{i+1}=-\frac{1}{2}R^{-1}g^T\nabla V_i, \quad \omega_{i+1}=\frac{1}{2}\gamma^{-2}k^T\nabla V_i \quad (5)$$

Theorem 1: Suppose Assumption 1 holds and the convergence conditions of Kantorovtich's Theorem [53], [54] are satisfied consistently. Iterating on (4) and (5) is equivalent to a Newton method for solving the HJI equation (3), and the sequence $\{V_i\}_{i=0}^{\infty}$ converges to the optimal value V^* as $i\to\infty$.

Proof: Following similar logic of [26], this can be proved. ■

Remark 1: The Lyapunov equation (4) is a linear PDE with respect to V_i. Iterating on (4) is more feasible than solving the HJI equation directly. Note that the iteration formulas require complete knowledge of the dynamics. Some algorithms like [26], [27], [28] use integral reinforcement learning to remove the dependence on f, but the other dynamics or identification of dynamics are still necessary.

III. A Neural-Network Based Iterative ADP Algorithm

Now suppose two arbitrary policies u and ω are executed, and they stabilize the system (1) in a compact region. Denote u_i and ω_i as the results of (4) and (5) at the i-th iteration. They are assumed to be known at the $(i+1)$-th iteration and are further used to compute V_i, u_{i+1}, and ω_{i+1}. Along the dynamics solutions, the time derivative of V_i equals $\dot{V}_i=\nabla V_i^T(f+gu+k\omega)$. Subtract (4) from $\dot{V}_i$ and substitute (5)

$$\dot{V}_i=-2u_{i+1}^TR(u-u_i)+2\gamma^2\omega_{i+1}^T(\omega-\omega_i)-r(x,u_i,\omega_i)$$

Based on the idea of integral reinforcement learning, integrate both sides between t and t' $(t'>t)$ and we have

$$0=V_i(x(t'))-V_i(x(t))+\int_t^{t'}2u_{i+1}^TR(u-u_i)d\tau - \int_t^{t'}2\gamma^2\omega_{i+1}^T(\omega-\omega_i)d\tau+\int_t^{t'}r(x,u_i,\omega_i)d\tau \quad (6)$$

Remark 2: Compared to (4) and (5) that are formulated using system dynamics, (6) is completely model-free. To solve (6) for V_i, u_{i+1}, and ω_{i+1}, three NNs (actor, disturber, and critic) are constructed to approximate the control and disturbance policies and the value.

According to the Weirstrass high-order approximation theorem [55], [56], a smooth function can be uniformly approximated on a compact set by neural networks. On the compact set Ω, we let[1]

$$V_i(x)=c_{1,i+1}^T\phi_1(x)+\varepsilon_{1,i+1}(x)$$
$$u_{i+1}(x)=c_{2,i+1}^T\phi_2(x)+\varepsilon_{2,i+1}(x)$$
$$\omega_{i+1}(x)=c_{3,i+1}^T\phi_3(x)+\varepsilon_{3,i+1}(x)$$

and

$$u_i(x)=c_{2,i}^T\phi_2(x)+\varepsilon_{2,i}(x)$$
$$\omega_i(x)=c_{3,i}^T\phi_3(x)+\varepsilon_{3,i}(x)$$

where $\phi_1:\mathbb{R}^n\to\mathbb{R}^{K_1}$, $\phi_2:\mathbb{R}^n\to\mathbb{R}^{K_2}$, $\phi_3:\mathbb{R}^n\to\mathbb{R}^{K_3}$ denote linearly independent basis functions, $c_{1,\bullet}\in\mathbb{R}^{K_1}$, $c_{2,\bullet}\in\mathbb{R}^{K_2\times m}$, $c_{3,\bullet}\in\mathbb{R}^{K_3\times q}$ are the coefficient vector and matrices, $\varepsilon_{1,\bullet}$, $\varepsilon_{2,\bullet}$, $\varepsilon_{3,\bullet}$ are reconstruction errors with appropriate dimensions. K_1, K_2, and K_3 indicate the numbers of hidden-layer neurons. Assume basis functions, coefficients and reconstruction errors are all bounded on Ω. Furthermore, when $K_1\to\infty, K_2\to\infty, K_3\to\infty$, we have $\varepsilon_{1,\bullet}\to 0, \varepsilon_{2,\bullet}\to 0, \varepsilon_{3,\bullet}\to 0$.

After substituting the above NNs into (6), we obtain

$$\begin{aligned}\varepsilon_L=&\big(\phi_1(x(t'))-\phi_1(x(t))\big)^Tc_{1,i+1}\\&+\int_t^{t'}2\phi_2^Tc_{2,i+1}R(u-c_{2,i}^T\phi_2)d\tau\\&-\int_t^{t'}2\gamma^2\phi_3^Tc_{3,i+1}(\omega-c_{3,i}^T\phi_3)d\tau\\&+\int_t^{t'}r(x,c_{2,i}^T\phi_2,c_{3,i}^T\phi_3)d\tau\end{aligned}$$

where ε_L is the Lyapunov error due to NN reconstruction errors, denoted by

$$\begin{aligned}\varepsilon_L\equiv&-\varepsilon_{1,i+1}(x(t'))+\varepsilon_{1,i+1}(x(t))\\&-\int_t^{t'}2\big((u-c_{2,i}^T\phi_2)^TR\varepsilon_{2,i+1}-\phi_2^Tc_{2,i+1}R\varepsilon_{2,i}\\&\qquad-\varepsilon_{2,i+1}^TR\varepsilon_{2,i}\big)d\tau\\&+\int_t^{t'}2\gamma^2\big((\omega-c_{3,i}^T\phi_3)^T\varepsilon_{3,i+1}-\phi_3^Tc_{3,i+1}\varepsilon_{3,i}\\&\qquad-\varepsilon_{3,i+1}^T\varepsilon_{3,i}\big)d\tau\\&-\int_t^{t'}(2\phi_2^Tc_{2,i}R\varepsilon_{2,i}+\varepsilon_{2,i}^TR\varepsilon_{2,i})d\tau\\&+\int_t^{t'}\gamma^2(2\phi_3^Tc_{3,i}\varepsilon_{3,i}+\varepsilon_{3,i}^T\varepsilon_{3,i})d\tau\end{aligned}$$

To identify the ideal coefficients, we use $W_{1,i+1}$, $W_{2,i+1}$, and $W_{3,i+1}$ to estimate $c_{1,i+1}$, $c_{2,i+1}$, and $c_{3,i+1}$. The NN approximators are parameterized as

$$\hat{V}_i(x)=W_{1,i+1}^T\phi_1(x)$$
$$\hat{u}_{i+1}(x)=W_{2,i+1}^T\phi_2(x)$$

[1]To denote consistently, the NN coefficients of V_i use subscript $(i+1)$. The rest values below follow the same notation.

$$\hat{\omega}_{i+1}(x) = W_{3,i+1}^T \phi_3(x)$$

Assume the NN weights $W_{2,i}$ and $W_{3,i}$, with respect to the given policies u_i and ω_i, are already known. With the system data, $W_{1,i+1}, W_{2,i+1}, W_{3,i+1}$ can be solved by the least-squares (LS) method. Given a strictly increasing time sequence $\{t_k\}_{k=0}^{l}$, for each interval, define the residual error e_k as

$$\begin{aligned} e_k =& \big(\phi_1(x(t_{k+1})) - \phi_1(x(t_k))\big)^T W_{1,i+1} \\ &+ \int_{t_k}^{t_{k+1}} 2\phi_2^T W_{2,i+1} R(u - W_{2,i}^T \phi_2) d\tau \\ &- \int_{t_k}^{t_{k+1}} 2\gamma^2 \phi_3^T W_{3,i+1}(\omega - W_{3,i}^T \phi_3) d\tau \\ &+ \int_{t_k}^{t_{k+1}} r(x, W_{2,i}^T \phi_2, W_{3,i}^T \phi_3) d\tau \end{aligned} \tag{7}$$

By the Kronecker product $\otimes$, we have

$$\begin{aligned} &\phi_2^T W_{2,i+1} R(u - W_{2,i}^T \phi_2) = \\ &\qquad \big((u - W_{2,i}^T \phi_2)^T R \otimes \phi_2^T\big) \mathbf{v}(W_{2,i+1}) \\ &\phi_3^T W_{3,i+1}(\omega - W_{3,i}^T \phi_3) = \big((\omega - W_{3,i}^T \phi_3)^T \otimes \phi_3^T\big) \mathbf{v}(W_{3,i+1}) \end{aligned}$$

where $\mathbf{v}(\cdot)$ is a vector operator which transforms a matrix into a vector by stacking its columns. Then, (7) can be rewritten as

$$e_k = \theta_k^T(\bar{W}_i)\bar{W}_{i+1} + \xi(\bar{W}_i)$$

where $W_{1,i+1}, W_{2,i+1}, W_{3,i+1}$ are integrated into the vector $\bar{W}_{i+1} = [W_{1,i+1}^T, \mathbf{v}(W_{2,i+1})^T, \mathbf{v}(W_{3,i+1})^T]^T \in \mathbb{R}^{\bar{K}}$ and $\bar{K} = K_1 + mK_2 + qK_3$. $\bar{W}_i$ is defined in the same way by $W_{1,i}$, $W_{2,i}$, and $W_{3,i}$. θ_k and ξ_k are defined as

$$\theta_k(\bar{W}_i) = \begin{bmatrix} \phi_1(x(t_{k+1})) - \phi_1(x(t_k)) \\ \int_{t_k}^{t_{k+1}} 2R(u - W_{2,i}^T \phi_2) \otimes \phi_2 d\tau \\ -\int_{t_k}^{t_{k+1}} 2\gamma^2 (\omega - W_{3,i}^T \phi_3) \otimes \phi_3 d\tau \end{bmatrix} \in \mathbb{R}^{\bar{K}}$$

$$\xi_k(\bar{W}_i) = \int_{t_k}^{t_{k+1}} r(x, W_{2,i}^T \phi_2, W_{3,i}^T \phi_3) d\tau \in \mathbb{R}$$

The goal is to find a group of weights that minimize residual errors in the LS sense

$$\min_{\bar{W}_{i+1}} \sum_{k=0}^{l-1} e_k^2$$

Assumption 2 (Persistency of excitation (PE)): For each $i \geq 0$, there exist $l_0 > 0$ and $\delta > 0$ such that for all $l \geq l_0$,

$$\frac{1}{l} \sum_{k=0}^{l-1} \theta_k(\bar{W}_i) \theta_k^T(\bar{W}_i) \geq \delta I_{\bar{K}}$$

where $I_{\bar{K}}$ is the identity matrix with a given size.

According to the definition of $\theta_k(\bar{W}_i)$, to guarantee the PE condition, u and ω need to be sufficiently different from $\hat{u}_{i+1}$ and $\hat{\omega}_{i+1}$, and states need to be persistently excited. So, u and ω are designed to be probing and exploratory. The LS solution has

$$\bar{W}_{i+1} = -\big(\Theta^T(\bar{W}_i)\Theta(\bar{W}_i)\big)^{-1} \Theta^T(\bar{W}_i)\Xi(\bar{W}_i) \tag{8}$$

where

$$\Theta(\bar{W}_i) = [\theta_0(\bar{W}_i), \ldots, \theta_{l-1}(\bar{W}_i)]^T \tag{9}$$

$$\Xi(\bar{W}_i) = [\xi_0(\bar{W}_i), \ldots, \xi_{l-1}(\bar{W}_i)]^T \tag{10}$$

Once a new $\bar{W}_{i+1}$ is calculated, it replaces $\bar{W}_i$ and start the next iteration.

We propose a NNs-based iterative ADP algorithm for solving the unknown nonlinear ZSG. With initial policy weights given, the critic, actor and disturber NN weights are updated by iterating on (8). Under the PE condition, the algorithm doesn't need dynamics. It is interesting to note that the critic weights $W_{1,i}$ play an intermediate role in the learning process. Even though they determine the LS solution, they are not needed afterward.

In our algorithm, the control and disturbance policies and values are approximated by NNs and weights are solved by the LS method. So the original convergence conclusion needs to be reconsidered.

IV. Theoretical Analysis

A. Convergence of the iterative ADP algorithm

By iterating on (8), a sequence of NN weights is obtained. Since there exists a solution to the HJI equation, let the optimal value and saddle point policies be represented by NNs in the form

$$V^*(x) = c_{1,*}^T \phi_1(x) + \varepsilon_{1,*}(x) \tag{11}$$

$$u^*(x) = c_{2,*}^T \phi_2(x) + \varepsilon_{2,*}(x) \tag{12}$$

$$\omega^*(x) = c_{3,*}^T \phi_3(x) + \varepsilon_{3,*}(x) \tag{13}$$

Now consider V^*, u^*, ω^* and use the IRL technique. A similar equation is derived as (6) after making some manipulations

$$\begin{aligned} 0 = V^*(x(t')) - V^*(x(t)) &+ \int_t^{t'} {u^*}^T R(2u - u^*) d\tau \\ &- \int_t^{t'} \gamma^2 {\omega^*}^T (2\omega - \omega^*) d\tau + \int_t^{t'} h^T h d\tau \end{aligned}$$

After substituting (11)–(13), it becomes

$$\begin{aligned} \varepsilon_{HJI} =& \big(\phi_1(x(t')) - \phi_1(x(t))\big)^T c_{1,*} \\ &+ \int_t^{t'} \phi_2^T c_{2,*} R(2u - c_{2,*}^T \phi_2) d\tau \\ &- \int_t^{t'} \gamma^2 \phi_3^T c_{3,*}(2\omega - c_{3,*}^T \phi_3) d\tau + \int_t^{t'} h^T h d\tau \end{aligned}$$

where ε_{HJI} is the HJI error with

$$\begin{aligned} \varepsilon_{HJI} \equiv& -\varepsilon_{1,*}(x(t')) + \varepsilon_{1,*}(x(t)) \\ &- \int_t^{t'} \big(2(u - c_{2,*}^T \phi_2)^T R \varepsilon_{2,*} - \varepsilon_{2,*}^T R \varepsilon_{2,*}\big) d\tau \\ &+ \int_t^{t'} \gamma^2 \big(2(\omega - c_{3,*}^T \phi_3)^T \varepsilon_{3,*} - \varepsilon_{3,*}^T \varepsilon_{3,*}\big) d\tau \end{aligned}$$

When the ideal values of $c_{1,*}$, $c_{2,*}$, and $c_{3,*}$ are determined, the approximate optimal value and saddle point are acquired. Denote $W_{1,*}, W_{2,*}, W_{3,*}$ as their estimations. Along

the dynamics solutions between the time series $\{t_k\}_{k=0}^{l}$, the estimated weights define a group of residual errors

$$\begin{aligned} d_k =& \big(\phi_1(x(t_{k+1})) - \phi_1(x(t_k))\big)^T W_{1,*} \\ &+ \int_{t_k}^{t_{k+1}} \phi_2^T W_{2,*} R(2u - W_{2,*}^T \phi_2) d\tau \\ &- \int_{t_k}^{t_{k+1}} \gamma^2 \phi_3^T W_{3,*}(2\omega - W_{3,*}^T \phi_3) d\tau + \int_{t_k}^{t_{k+1}} h^T h d\tau \end{aligned}$$

Now, the problem becomes a nonlinear least-squares problem (NLSP). The optimal weights correspond to parameters that minimize the square error

$$\min_{\bar{W}_*} D^T(\bar{W}_*) D(\bar{W}_*) \tag{14}$$

where $\bar{W}_* = [W_{1,*}^T, \mathbf{v}(W_{2,*})^T, \mathbf{v}(W_{3,*})^T]^T \in \mathbb{R}^{\bar{K}}$, and $D(\bar{W}_*)$ is the residual error vector $D(\bar{W}_*) = [d_0, \ldots, d_{l-1}]^T \in \mathbb{R}^l$. It has been demonstrated that Gauss-Newton method can solve this optimization problem. Next, lemmas will reveal the connection between Gauss-Newton method and our iterative ADP algorithm.

Lemma 1: The Jacobian matrix $\mathbf{J} \in \mathbb{R}^{l*\bar{K}}$ of NLSP with respect to (14) is defined as

$$(\mathbf{J}(\bar{W}_*))_{ij} = \frac{\partial (D(\bar{W}_*))_i}{\partial (\bar{W}_*)_j}$$

When substituting $\bar{W}_*$ into (9), we have $\mathbf{J}(\bar{W}_*) = \Theta(\bar{W}_*)$.

Proof: The partial derivatives of d_k toward the NN weights are

$$\begin{aligned} \frac{\partial d_k}{\partial W_{1,*}} &= \phi_1(x(t_{k+1})) - \phi_1(x(t_k)) \\ \frac{\partial d_k}{\partial W_{2,*}} &= \int_{t_k}^{t_{k+1}} 2(\phi_2 u^T R - \phi_2 \phi_2^T W_{2,*} R) d\tau \\ \frac{\partial d_k}{\partial W_{3,*}} &= -\int_{t_k}^{t_{k+1}} 2\gamma^2 (\phi_3 \omega^T - \phi_3 \phi_3^T W_{3,*}) d\tau \end{aligned}$$

According to the Kronecker product representation, it is concluded that $\partial d_k / \partial \bar{W}_* = \theta_k(\bar{W}_*)$. Hence $\mathbf{J}(\bar{W}_*) = \Theta(\bar{W}_*)$. ∎

Lemma 2: Given a parametric vector $\bar{W}_i \in \mathbb{R}^{\bar{K}}$, if Assumption 2 is continually satisfied, the calculation of $\bar{W}_{i+1}$ based on (8) is equivalent to the Gauss-Newton equation

$$\bar{W}_{i+1} = \bar{W}_i - \big(\mathbf{J}^T(\bar{W}_i)\mathbf{J}(\bar{W}_i)\big)^{-1}\mathbf{J}^T(\bar{W}_i) D(\bar{W}_i) \tag{15}$$

Proof: From Lemma 1 and the definitions of Θ and Ξ,

$$\mathbf{J}(\bar{W}_i)\bar{W}_i - D(\bar{W}_i) = -\Xi(\bar{W}_i)$$

After substituting into (8), it is deduced that

$$\begin{aligned} \bar{W}_{i+1} =& \big(\mathbf{J}^T(\bar{W}_i)\mathbf{J}(\bar{W}_i)\big)^{-1}\mathbf{J}^T(\bar{W}_i)\big(\mathbf{J}(\bar{W}_i)\bar{W}_i - D(\bar{W}_i)\big) \\ =& \bar{W}_i - \big(\mathbf{J}^T(\bar{W}_i)\mathbf{J}(\bar{W}_i)\big)^{-1}\mathbf{J}^T(\bar{W}_i) D(\bar{W}_i) \end{aligned}$$

∎

According to the above analysis, our algorithm is actually a Gauss-Newton method for solving the optimization problem (14). The convergence theorem is presented.

Theorem 2: Suppose the following assumptions hold

1) Assumption 2 holds continually;
2) there exist $\bar{W}_* \in \mathbb{R}^{\bar{K}}$ such that $\mathbf{J}^T(\bar{W}_*) D(\bar{W}_*) = 0$;
3) the Jacobian matrix $\mathbf{J}(\bar{W}_*)$ at $\bar{W}_*$ has full rank $\bar{K}$;
4) $\rho\Big(\big(\mathbf{J}^T(\bar{W}_*)\mathbf{J}(\bar{W}_*)\big)^{-1}\big(\sum_{i=1}^{l} D_i(\bar{W}_*)\nabla^2 D_i(\bar{W}_*)\big)\Big) < 1$, where $\rho(A)$ indicates the spectral radius of a square matrix A and ∇^2 is the Hessian matrix.

Under the above assumptions, there exists $\varepsilon > 0$ such that the sequence $\{\bar{W}_i\}$ generated by (8) converges to $\bar{W}_*$ for all $\bar{W}_0 \in \mathbb{D} \equiv \{\bar{W} | \|\bar{W} - \bar{W}_*\| < \varepsilon\}$ [2].

Proof: According to [57], [58], if the assumptions in the theorem are satisfied, the Gauss-Newton method (15) converges to $\bar{W}_*$. Combined with Lemma 2, the proof is complete. ∎

Remark 3: The first requirement in Theorem 2 reveals the importance of the PE condition. One approach to guarantee the PE condition is to add probing noise to control inputs. In the simulations, we let $u = u' + e_u$ and $\omega = \omega' + e_\omega$. u' and ω' are policies for which $(f + gu' + k\omega')$ is stable, and e_u and e_ω are probing noise.

Remark 4: From the viewpoint of RL, the PE condition is equivalent to the infinitely-often-visited condition required by the famous Q-learning algorithm [34], [59]. Both of them suggest system observations shall contain as much dynamical information as possible, not just performing the current learned policies.

Remark 5: Another sufficient condition for Theorem 2 emphasizes that the initial $\bar{W}_0$ shall be within the region $\mathbb{D}$. In PI-based ADP algorithms, such initial condition is widely considered. In [20], [21], the inner loop was started from a disturbance $\omega_0 = 0$ and the outer loop was initialized to a stabilizing controller u_0. In [26], a requirement for the initial value is imposed. In this paper, we initialize $W_{2,0}$ and $W_{3,0}$ of the actor and disturber NNs to values such that $\hat{u}_0$ is a stabilizing control policy and $\hat{\omega}_0 = 0$. As for $W_{1,0}$ of the critic, since it is not needed in the calculation of (8), it is neglected in the initialization.

B. Uniform convergence to the HJI solution

In this part, we will prove that the approximators with the LS solutions uniformly approximate V_i, u_i, ω_i as defined by (4) and (5). Then, we will conclude that the HJI solution is uniformly convergent.

Lemma 3: Under Assumption 2, for each $i \geq 0$

$$\begin{aligned} \lim_{K_1,K_2,K_3\to\infty} \hat{V}_i(x) &= V_i(x) \\ \lim_{K_1,K_2,K_3\to\infty} \hat{u}_{i+1}(x) &= u_{i+1}(x) \\ \lim_{K_1,K_2,K_3\to\infty} \hat{\omega}_{i+1}(x) &= \omega_{i+1}(x) \end{aligned}$$

Proof: See the Appendix. ∎

Theorem 3: Under Assumptions 1 and 2, for arbitrary $\epsilon > 0$, there exist $i^* > 0$, $K_1^* > 0$, $K_2^* > 0$, $K_3^* > 0$, such that for all $x \in \Omega$

$$\begin{aligned} \big|\hat{V}_i(x) - V^*(x)\big| &\leq \epsilon \\ \big\|\hat{u}_{i+1}(x) - u^*(x)\big\| &\leq \epsilon \end{aligned}$$

[2]Throughout this paper, we use $|\cdot|$ as the magnitude of a scalar, $\|\cdot\|$ as the vector norm of a vector, and $\|\cdot\|_2$ as the induced matrix 2-norm.

$$\left\|\hat{\omega}_{i+1}(x) - \omega^*(x)\right\| \leq \epsilon$$

if $i > i^*$, $K_1 > K_1^*$, $K_2 > K_2^*$, $K_3 > K_3^*$.

Proof: The theorem is proved by Theorem 1 and Lemma 3. ■

Remark 6: From Theorems 2 and 3, when applying the NN-based iterative ADP algorithm to the nonlinear ZSG, the results not only converge, but also converge uniformly to the optimal value and saddle point.

V. Algorithm Implementation

This section is how to implement the iterative ADP algorithm to solve the unknown nonlinear ZSG with only online data. Before implementation, we need to transform the main equation (8) into another form. According to the Kronecker product representation, $\Theta(\bar{W}_i)$ and $\Xi(\bar{W}_i)$ are defined by (9) and (10), and can be rewritten as

$$\begin{aligned}\Theta(\bar{W}_i) = \big[&\delta_1, 2\delta_2(R \otimes I_{K_2}) - 2\delta_3(W_{2,i}R \otimes I_{K_2}),\\ &-2\gamma^2\delta_4 + 2\gamma^2\delta_5(W_{3,i} \otimes I_{K_3})\big]\\ \Xi(\bar{W}_i) = \big[&\delta_6 + \delta_3\mathbf{v}(W_{2,i}RW_{2,i}^T) - \gamma^2\delta_5\mathbf{v}(W_{3,i}W_{3,i}^T)\big]\end{aligned}$$

where

$$\begin{aligned}\delta_1 &= \big[\phi_1(x(t_1)) - \phi_1(x(t_0)), \ldots, \phi_1(x(t_l)) - \phi_1(x(t_{l-1}))\big]^T\\ \delta_2 &= \big[\int_{t_0}^{t_1} u \otimes \phi_2 d\tau, \ldots, \int_{t_{l-1}}^{t_l} u \otimes \phi_2 d\tau\big]^T\\ \delta_3 &= \big[\int_{t_0}^{t_1} \phi_2 \otimes \phi_2 d\tau, \ldots, \int_{t_{l-1}}^{t_l} \phi_2 \otimes \phi_2 d\tau\big]^T\\ \delta_4 &= \big[\int_{t_0}^{t_1} \omega \otimes \phi_3 d\tau, \ldots, \int_{t_{l-1}}^{t_l} \omega \otimes \phi_3 d\tau\big]^T\\ \delta_5 &= \big[\int_{t_0}^{t_1} \phi_3 \otimes \phi_3 d\tau, \ldots, \int_{t_{l-1}}^{t_l} \phi_3 \otimes \phi_3 d\tau\big]^T\\ \delta_6 &= \big[\int_{t_0}^{t_1} h^T h d\tau, \ldots, \int_{t_{l-1}}^{t_l} h^T h d\tau\big]^T\end{aligned}$$

The above matrices depend on system input data u and ω. These matrices can be used repeatedly to update Θ and Ξ at each iteration with new NN weights, which help to reduce the online interaction with the system.

The flowchart of the iterative ADP algorithm is given in Fig. 1. The process includes two phases. In the first *measurement phase*, it executes probing control and disturbance inputs on the system and collects online data. After a sufficient time, the algorithm is switched to the *learning phase*. The critic, actor, and disturber NN weights are trained offline iteratively until reaching the convergence. If the process does not converge, it will go back to the first phase and collect more data. In the end, the converged actor provides an H_∞ controller for the system.

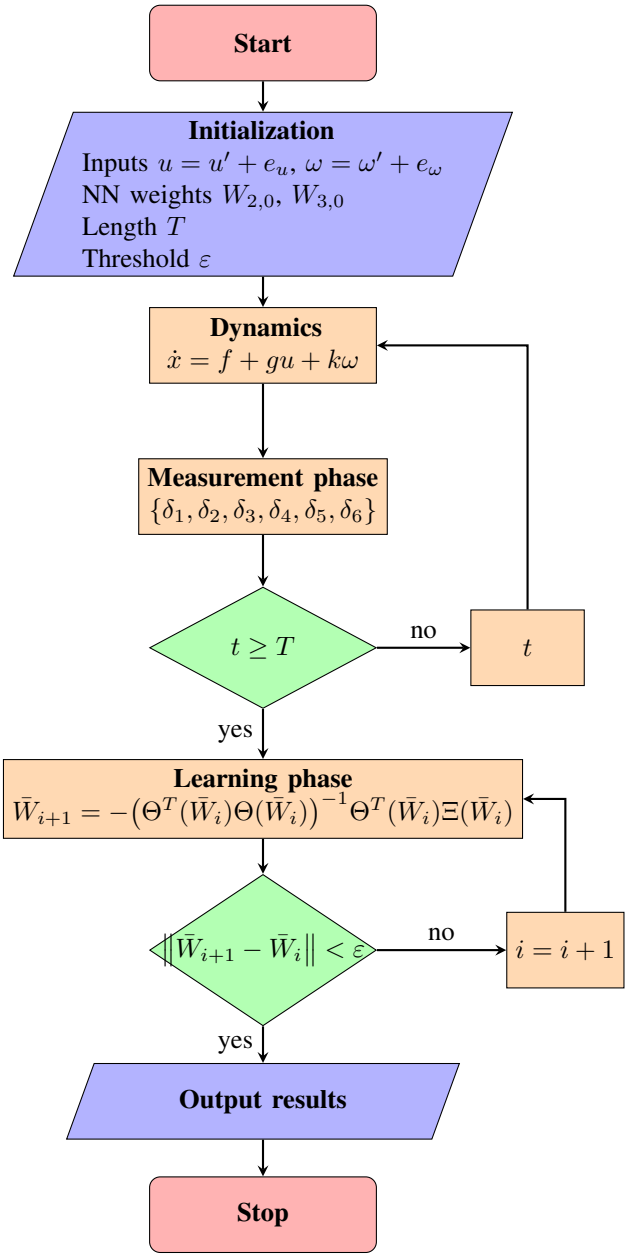


Fig. 1. The flowchart of the iterative ADP algorithm.

VI. Simulation Study

A. Example 1

First, we use the converse optimal control method [60] to design a nonlinear system with

$$\begin{aligned}f(x) &= \begin{bmatrix} -x_1 + x_2 \\ -x_1^3 - x_2^3 + \frac{1}{4}(x_1 + x_2)^2 x_2 - \frac{1}{4\gamma^2}x_1^2 x_2 \end{bmatrix}\\ g(x) &= \begin{bmatrix} 0 \\ x_1 + x_2 \end{bmatrix}, \quad k(x) = \begin{bmatrix} 0 \\ x_1 \end{bmatrix}\end{aligned}$$

Select $h(x) = [x_1^2, x_2^2]^T$, $R = 1$, and $\gamma = 4$. The optimal value function has

$$V^*(x) = \frac{1}{4}x_1^4 + \frac{1}{2}x_2^2$$

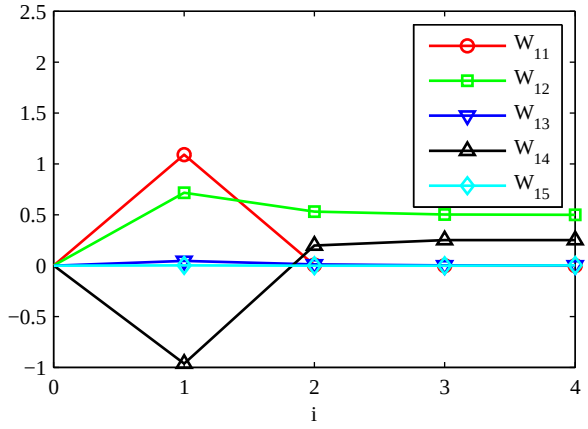


Fig. 2. NN weights of the critic in the learning phase of Example 1.

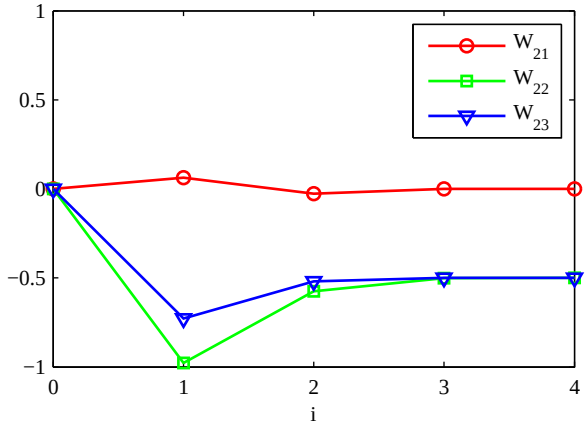


Fig. 3. NN weights of the actor in the learning phase of Example 1.

and the saddle point has

$$u^*(x) = -\frac{1}{2}x_1x_2 - \frac{1}{2}x_2^2, \quad \omega^*(x) = \frac{1}{32}x_1x_2$$

The basis function vectors for the critic, actor, and disturber NNs select

$$\phi_1(x) = [x_1^2, x_2^2, x_1^2x_2^2, x_1^4, x_2^4]^T$$
$$\phi_2(x) = [x_1^2, x_1x_2, x_2^2]^T$$
$$\phi_3(x) = [x_1^2, x_1x_2, x_2^2]^T$$

So the ideal coefficients are $c_1 = [0, 0.5, 0, 0.25, 0]^T$, $c_2 = [0, -0.5, -0.5]^T$, and $c_3 = [0, 0.03125, 0]^T$.

Since the system is self-stable, we choose probing control inputs $u = 10 * \big(\sin(10t) + \sin(9.3t) + \sin(5.2t) + 3.02\big)$ and $\omega = 10 * \big(\sin(11t) + \sin(7.8t) + \sin(9.5t) - 5.78\big)$. We let the system start from $x(0) = [1, -1]^T$ with an integral time of 0.1s. After 3s, the algorithm will terminate the measurement phase and switch to the learning phase. The initial weights for the actor and disturber NNs are $W_{2,0} = [0, 0, 0]^T$, $W_{3,0} = [0, 0, 0]^T$. The convergence threshold is 10^{-6}. The iterative ADP algorithm converges at the 4th iteration with outputs $W_{1,4} = [-0.0000, 0.5000, 0.0000, 0.2500, 0.0000]^T$, $W_{2,4} = [-0.0000, -0.5000, -0.5000]^T$, $W_{3,4} = [0.0000, 0.03215, 0.0000]^T$. The NN weights are depicted in Figs. 2–4. After 3s, the converged actor and disturber replace the probing control inputs. The whole state and control trajectories are shown in Figs. 5 and 6.

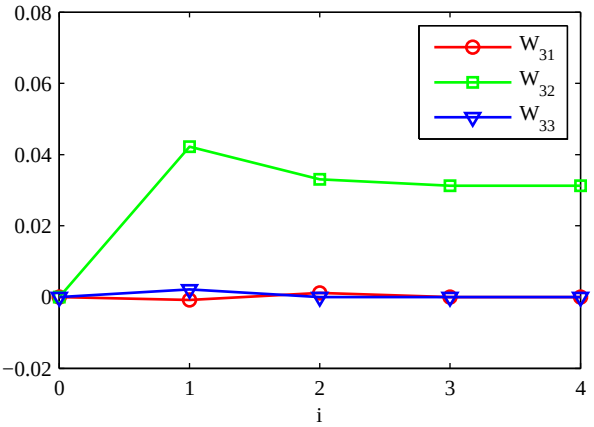


Fig. 4. NN weights of the disturber in the learning phase of Example 1.

B. Example 2

The second experiment uses a nonlinear system from [28]. Two online ADP algorithms, CRLA and SPIA, have solved the nonlinear ZSG problem. The dynamics is

$$\dot{x} = \begin{bmatrix} -x_1 + x_2 \\ -0.5 * (x_1 + x_2) + 0.5x_2 \sin(x_1)^2 \end{bmatrix} + \begin{bmatrix} 0 \\ \sin(x_1) \end{bmatrix} u + \begin{bmatrix} 0 \\ \cos(x_1) \end{bmatrix} \omega$$

We select $h(x) = [x_1, x_2]^T$, $R = 1$, and $\gamma = 2$. Note that there is no analytic solution to the problem. We select up to

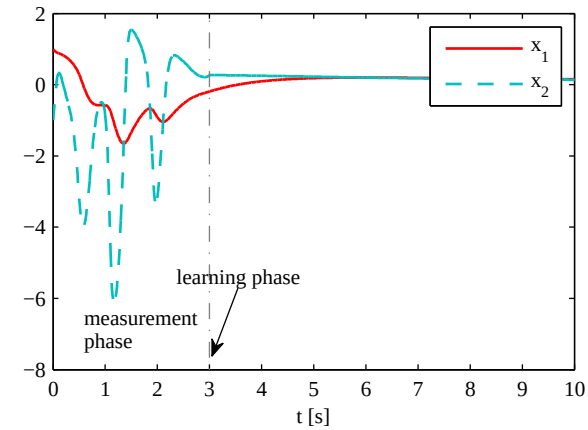


Fig. 5. Trajectories of state variables in Example 1.

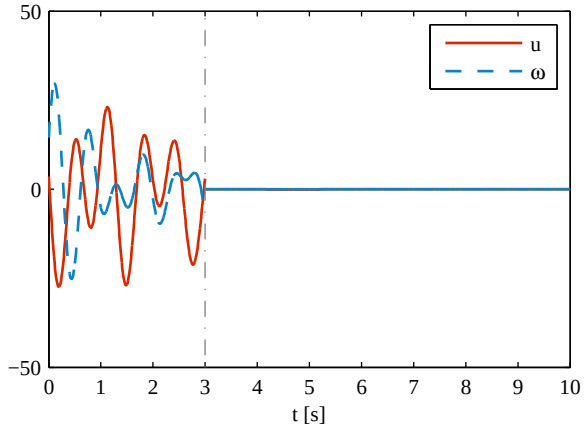


Fig. 6. Trajectories of control variables in Example 1.

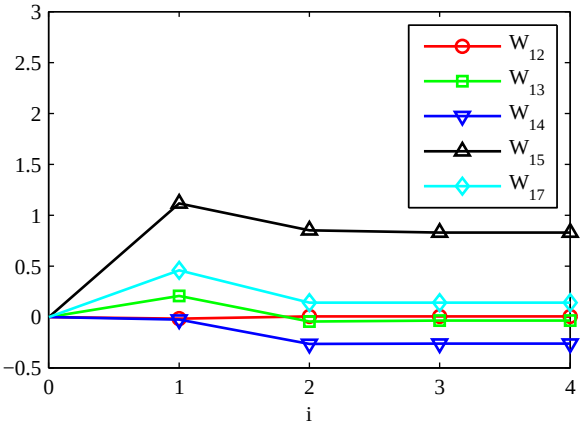


Fig. 7. Some weights of the critic in the learning phase of Example 2.

fourth order polynomials to define the basis functions of the critic, actor, and disturber NNs, i.e.

$$\phi_1(x) = \phi_2(x) = \phi_3(x) = [x_1, x_2, x_1^2, x_1x_2, x_2^2, x_1^3, x_1^2x_2, x_1x_2^2, x_2^3, x_1^4, x_1^3x_2, x_1^2x_2^2, x_1x_2^3, x_2^4]^T$$

The total number of parameters is $\bar{K} = 42$. The experiment is simulated using the same setup as Example 1. We set Initial weights $W_{2,0}$ and $W_{3,0}$ to be 0. The measurement phase lasts 20s. After that the learning phase starts training the NNs. After 4 iterations, the algorithm reaches convergence. Figs. 7–9 show some weights in the critic, actor and disturber NNs. The final actor is formulated as

$$\hat{u}_4(x) = [0.0230, 0.0109, 0.1931, -0.8625, 0.0019, -0.0786, -0.0498, 0.0082, -0.0025, -0.0447, 0.0831, -0.0118, 0.0054, -0.0013]\phi_2(x)$$

After the learning phase, the converged actor and disturber replace the probing control inputs. Figs. 10 and 11 show the whole trajectories of the system.

Our algorithm only requires a online measurement time of 20s, as compared to 270s of CRLA and more than 800s of SPIA. In addition, implementation of CRLA and SPIA

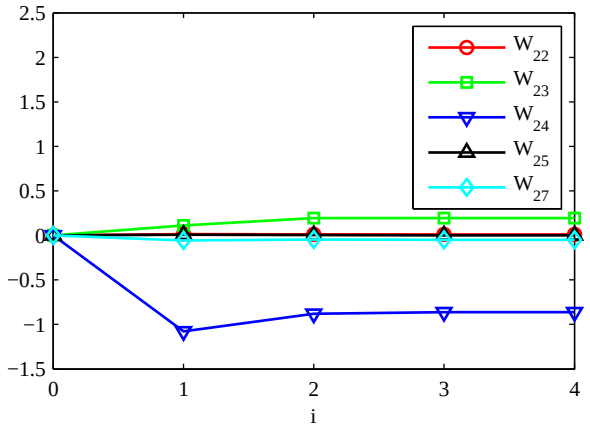


Fig. 8. Some weights of the actor in the learning phase of Example 2.

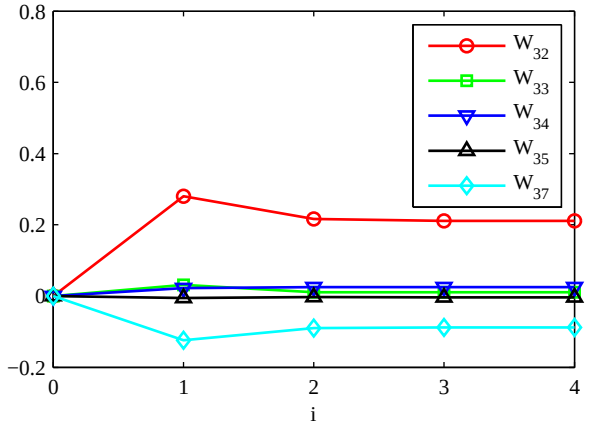


Fig. 9. Some weights of the disturber in the learning phase of Example 2.

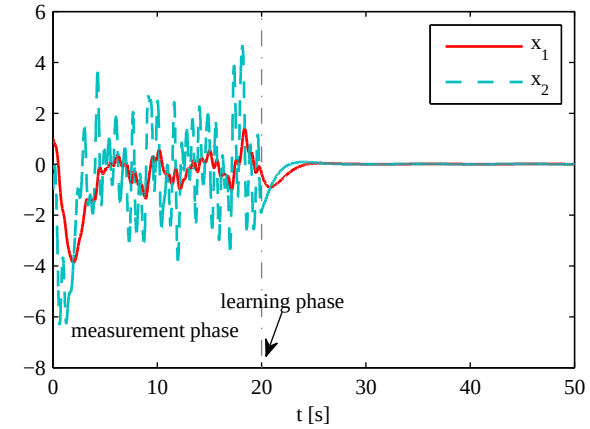


Fig. 10. Trajectories of state variables in Example 2.

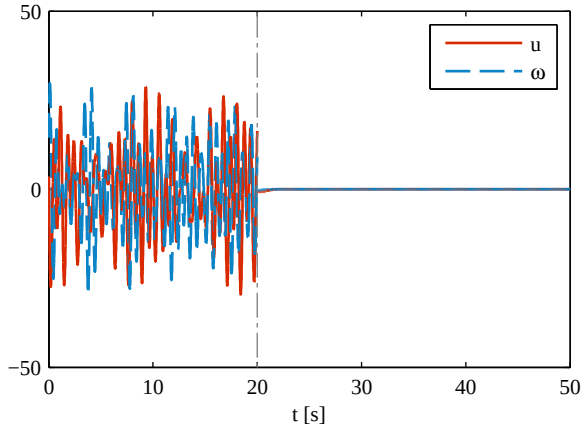


Fig. 11. Trajectories of control variables in Example 2.

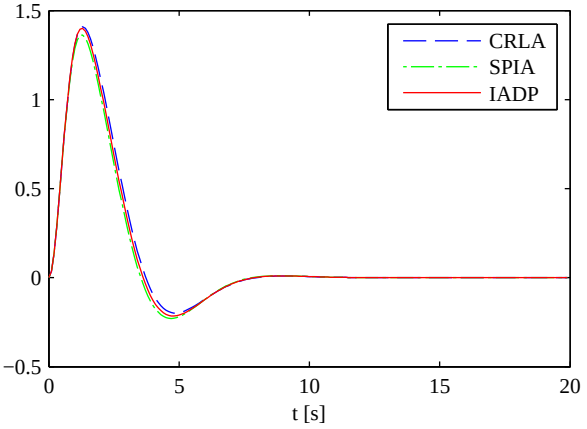


Fig. 12. Trajectories of x_1 for iterative ADP algorithm (IADP), concurrent reinforcement learning algorithm (CRLA), and synchronous policy iteration algorithm (SPIA).

relies on partial system dynamics, while implementation of our algorithm doesn't need any dynamics.

Next, our converged actor is compared with the results of CRLA and SPIA [28] in the same finite-energy run. The system is initially at rest, and disturbance $\omega(t) = 8\cos(t)\exp^{-t}$ is applied. Figs. 12–14 show trajectories of the state and control when executing three controllers separately. Fig. 15 shows the disturbance attenuation $\int_0^t (h^T h + u^T R u)d\tau / \int_0^t \omega^T \omega d\tau$ of the three runs. From these plots, performance differences of these three controllers are barely noticeable, except that CRLA has the best attenuation effect. Our controller performs similar to CRLA while SPIA performs the worst.

VII. Conclusion

In this paper, The continuous-time unknown nonlinear zero-sum game is approximately solved by an iterative ADP algorithm using online data. Data containing complete dynamical information of the system are utilized properly to learn the NN parameters of the control and disturbance policies and the value. The same data are used repeatedly at each iteration, which significantly reduces the measurement time.

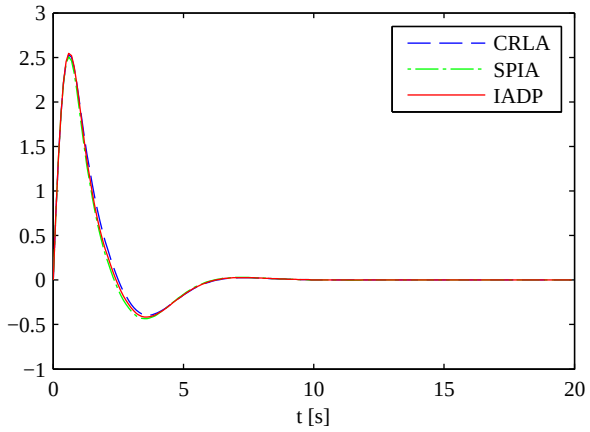


Fig. 13. Trajectories of x_2 for iterative ADP algorithm (IADP), concurrent reinforcement learning algorithm (CRLA), and synchronous policy iteration algorithm (SPIA).

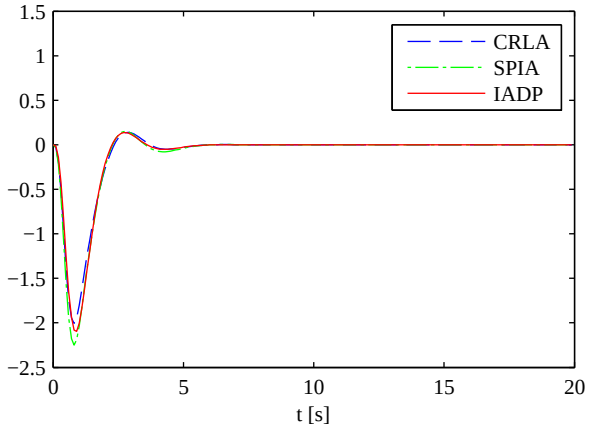


Fig. 14. Trajectories of u for iterative ADP algorithm (IADP), concurrent reinforcement learning algorithm (CRLA), and synchronous policy iteration algorithm (SPIA).

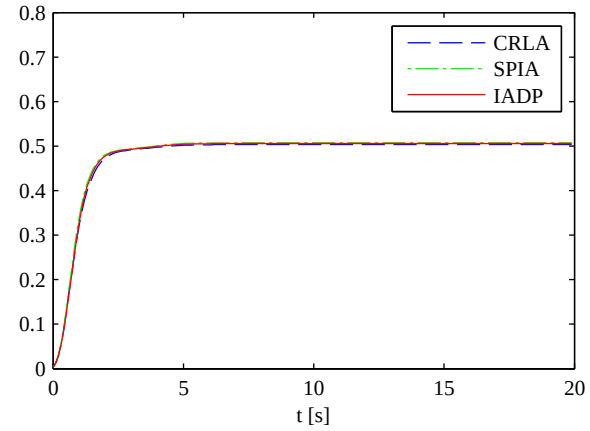


Fig. 15. Disturbance attenuation for iterative ADP algorithm (IADP), concurrent reinforcement learning algorithm (CRLA), and synchronous policy iteration algorithm (SPIA).

In the future, we will focus on applying this algorithm to more complex and realistic systems. It requires the appropriate selection of basis functions for neural networks and the delicate design of the learning process. Currently, our basis functions are selected manually. Motivated by literatures like [61], [62], we will further research on methods to automatically learn basis functions.

APPENDIX

Before proving Lemma 3, an auxiliary lemma is presented. Given $\hat{u}_i$ and $\hat{\omega}_i$, let $\tilde{V}_i$ be the solution of the Lyapunov equation

$$0 = r(x, \hat{u}_i, \hat{\omega}_i) + \nabla \tilde{V}_i^T \left(f + g\hat{u}_i + k\hat{\omega}_i\right), \tilde{V}_i(0) = 0$$

Define

$$\begin{aligned}\tilde{u}_{i+1} =& -\frac{1}{2}R^{-1}g^T\nabla\tilde{V}_i \\ \tilde{\omega}_{i+1} =& \frac{1}{2}\gamma^{-2}k^T\nabla\tilde{V}_i\end{aligned}$$

Lemma 4: Under the PE condition, for any $x \in \Omega$ we have

$$\begin{aligned}\lim_{K_1,K_2,K_3\to\infty} \hat{V}_i(x) &= \tilde{V}_i(x) \\ \lim_{K_1,K_2,K_3\to\infty} \hat{u}_{i+1}(x) &= \tilde{u}_{i+1}(x) \\ \lim_{K_1,K_2,K_3\to\infty} \hat{\omega}_{i+1}(x) &= \tilde{\omega}_{i+1}(x)\end{aligned}$$

Proof: Following the derivation of (6), a similar formula is obtained for $\tilde{V}_i$, $\tilde{u}_{i+1}$, $\tilde{\omega}_{i+1}$ with the time sequence $\{t_k\}_{k=0}^l$

$$\begin{aligned}0 =& \tilde{V}_i(x(t_{k+1})) - \tilde{V}_i(x(t_k)) + \int_{t_k}^{t_{k+1}} 2\tilde{u}_{i+1}^T R(u - \hat{u}_i)d\tau \\ &- \int_{t_k}^{t_{k+1}} 2\gamma^2\tilde{\omega}_{i+1}^T(\omega - \hat{\omega}_i)d\tau + \int_{t_k}^{t_{k+1}} r(x, \hat{u}_i, \hat{\omega}_i)d\tau\end{aligned} \tag{16}$$

Suppose $\tilde{V}_i(x) = \tilde{c}_{1,i+1}^T\phi_1(x) + \tilde{\varepsilon}_{1,i+1}(x)$, $\tilde{u}_{i+1}(x) = \tilde{c}_{2,i+1}^T\phi_2(x) + \tilde{\varepsilon}_{2,i+1}(x)$, and $\tilde{\omega}_{i+1}(x) = \tilde{c}_{3,i+1}^T\phi_3(x) + \tilde{\varepsilon}_{3,i+1}(x)$. Subtract (16) from (7)

$$e_k = \theta_k^T\Delta + \eta_k$$

where

$$\Delta = \begin{bmatrix} W_{1,i+1} - \tilde{c}_{1,i+1} \\ \mathbf{v}(W_{2,i+1} - \tilde{c}_{2,i+1}) \\ \mathbf{v}(W_{3,i+1} - \tilde{c}_{3,i+1}) \end{bmatrix}$$

and

$$\begin{aligned}\eta_k =& -\tilde{\varepsilon}_{1,i+1}(x(t_{k+1})) + \tilde{\varepsilon}_{1,i+1}(x(t_k)) \\ &- \int_{t_k}^{t_{k+1}} 2(u - \hat{u}_i)^T R\tilde{\varepsilon}_{2,i+1}d\tau \\ &+ \int_{t_k}^{t_{k+1}} 2\gamma^2(\omega - \hat{\omega}_i)^T\tilde{\varepsilon}_{3,i+1}d\tau\end{aligned}$$

Since NN weights are determined in the LS sense, we have

$$\sum_{k=0}^{l-1} e_k^2 \le \sum_{k=0}^{l-1} \eta_k^2$$

Furthermore,

$$\begin{aligned}\sum_{k=0}^{l-1} \Delta^T\theta_k\theta_k^T\Delta = \sum_{k=0}^{l-1}(e_k - \eta_k)^2 \le& \sum_{k=0}^{l-1} 2(e_k^2 + \eta_k^2) \\ \le& 4l \max_{0\le k<l} \eta_k^2\end{aligned}$$

Combined with the PE condition, Δ is bounded by

$$\|\Delta\|^2 \le \frac{4}{\delta}\max_{0\le k<l}\eta_k^2$$

According to the Weierstrass higher-order approximation theorem, as the numbers of hidden-layer neurons $K_1 \to \infty$, $K_2 \to \infty$, $K_3 \to \infty$, the approximation errors $\tilde{\varepsilon}_{1,i+1} \to 0$, $\tilde{\varepsilon}_{2,i+1} \to 0$, $\tilde{\varepsilon}_{3,i+1} \to 0$, and $\eta_k \to 0$ on Ω. For arbitrary $\epsilon > 0$, there exist $K_1^* > 0$, $K_2^* > 0$, $K_3^* > 0$, such that for any $x \in \Omega$

$$\begin{aligned}|\hat{V}_i(x) - \tilde{V}_i(x)| \le& \|W_{1,i+1} - \tilde{W}_{1,i+1}\|\|\phi_1(x)\| + |\tilde{\varepsilon}_{1,i+1}| \\ \le& \frac{\epsilon}{2} + \frac{\epsilon}{2} = \epsilon\end{aligned}$$

if $K_1 > K_1^*$, $K_2 > K_2^*$, $K_3 > K_3^*$. Similarly,

$$\begin{aligned}\|\hat{u}_{i+1}(x) - \tilde{u}_{i+1}(x)\| \le \epsilon \\ \|\hat{\omega}_{i+1}(x) - \tilde{\omega}_{i+1}(x)\| \le \epsilon\end{aligned}$$

The proof is complete. ■

Proof of Lemma 3: Use the induction:

1) If $i = 0$, $\hat{u}_0 = u_0$. So $\tilde{V}_0 = V_0$, $\tilde{u}_1 = u_1$, $\tilde{\omega}_1 = \omega_1$. From Lemma 4, the conclusion is true.
2) Suppose for some $i > 0$, $\lim_{K_1,K_2,K_3\to\infty}\hat{V}_{i-1}(x) = V_{i-1}(x)$, $\lim_{K_1,K_2,K_3\to\infty}\hat{u}_i(x) = u_i(x)$, $\lim_{K_1,K_2,K_3\to\infty}\hat{\omega}_i(x) = \omega_i(x)$. After subtracting (16) from (6) and using the NNs, we yield

$$\begin{aligned}\varepsilon_\Delta =& \big(\phi_1(x(t_{k+1})) - \phi_1(x(t))\big)^T(c_{1,i+1} - \tilde{c}_{1,i+1}) \\ &+ \int_{t_k}^{t_{k+1}} 2\phi_2^T(c_{2,i+1} - \tilde{c}_{2,i+1})R(u - \hat{u}_i)d\tau \\ &- \int_{t_k}^{t_{k+1}} 2\gamma^2\phi_3^T(c_{3,i+1} - \tilde{c}_{3,i+1})(\omega - \hat{\omega}_i)d\tau\end{aligned}$$

where

$$\varepsilon_\Delta \equiv -\int_{t_k}^{t_{k+1}} 2u_{i+1}^T R(\hat{u}_i - u_i)d\tau \tag{17}$$

$$+ \int_{t_k}^{t_{k+1}} 2\gamma^2\omega_{i+1}^T(\hat{\omega}_i - \omega_i)d\tau \tag{18}$$

$$- \int_{t_k}^{t_{k+1}} (u_i^T Ru_i - \hat{u}_i^T R\hat{u}_i)d\tau \tag{19}$$

$$+ \int_{t_k}^{t_{k+1}} \gamma^2(\omega_i^T\omega_i - \hat{\omega}_i^T\hat{\omega}_i)d\tau \tag{20}$$

$$- \big(\varepsilon_{1,i+1}(x(t_{k+1})) - \tilde{\varepsilon}_{1,i+1}(x(t_{k+1}))\big) \tag{21}$$

$$+ \big(\varepsilon_{1,i+1}(x(t_k)) - \tilde{\varepsilon}_{1,i+1}(x(t_k))\big) \tag{22}$$

$$- \int_{t_k}^{t_{k+1}} 2(u - \hat{u}_i)^T R(\varepsilon_{2,i+1} - \tilde{\varepsilon}_{2,i+1})d\tau \tag{23}$$

$$+ \int_{t_k}^{t_{k+1}} 2\gamma^2(\omega - \hat{\omega}_i)^T(\varepsilon_{3,i+1} - \tilde{\varepsilon}_{3,i+1})d\tau \tag{24}$$

According to the hypotheses, (17)–(20) $\to 0$. According to the approximation property of NNs, (21)–(24) $\to 0$ as $K_1, K_2, K_3 \to \infty$. Under Assumption 2 and $\varepsilon_\Delta \to 0$, we conclude

$$\lim_{K_1,K_2,K_3\to\infty} \|c_{1,i+1} - \tilde{c}_{1,i+1}\| = 0$$
$$\lim_{K_1,K_2,K_3\to\infty} \|c_{2,i+1} - \tilde{c}_{2,i+1}\|_2 = 0$$
$$\lim_{K_1,K_2,K_3\to\infty} \|c_{3,i+1} - \tilde{c}_{3,i+1}\|_2 = 0$$

In other words,

$$\lim_{K_1,K_2,K_3\to\infty} \tilde{V}_i(x) = V_i(x)$$
$$\lim_{K_1,K_2,K_3\to\infty} \tilde{u}_{i+1}(x) = u_{i+1}(x)$$
$$\lim_{K_1,K_2,K_3\to\infty} \tilde{\omega}_{i+1}(x) = \omega_{i+1}(x)$$

In the end

$$\left|\hat{V}_i(x) - V_i(x)\right| \le \left|\hat{V}_i(x) - \tilde{V}_i(x)\right| + \left|\tilde{V}_i(x) - V_i(x)\right|$$

According to Lemma 4, when $K_1, K_2, K_3 \to \infty$, we have $\hat{V}_i \to V_i$. Similarly, $\hat{u}_{i+1} \to u_{i+1}$, $\hat{\omega}_{i+1} \to \omega_{i+1}$.

By induction, the proof is complete. ■

REFERENCES

[1] F. L. Lewis and V. L. Syrmos, *Optimal Control*. New York: Wiley, 1995.

[2] D. P. Bertsekas, *Dynamic Programming and Optimal Control*. Belmont, MA: Athena Scientific, 1995.

[3] K. Zhou, J. C. Doyle, and K. Glover, *Robust and Optimal Control*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1996.

[4] A. Isidori and W. Kang, "H_∞ control via measurement feedback for general nonlinear systems," *IEEE Trans. Autom. Control*, vol. 40, no. 3, pp. 466–472, Mar 1995.

[5] A. J. Van der Schaft, *L2-Gain and Passivity Techniques in Nonlinear Control*, 1st ed. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1996.

[6] S. Tijs, *Introduction to Game Theory*. India: Hindustan Book Agency, 2003.

[7] T. Basar and P. Bernhard, *H_∞ Optimal Control and Related Minimax Design Problems*. Boston, MA: Birkhäuser, 1995.

[8] T. Basar and G. J. Olsder, *Dynamic Noncooperative Game Theory*, 2nd ed. Philadelphia, PA: SIAM, 1999.

[9] J. Huang and C.-F. Lin, "Numerical approach to computing nonlinear H_∞ control laws," *J. Guidance Control Dynamics*, vol. 18, no. 5, pp. 989–994, 1995.

[10] R. W. Beard and T. W. Mclain, "Successive Galerkin approximation algorithms for nonlinear optimal and robust control," *Int. J. Control*, vol. 71, no. 5, pp. 717–743, 1998.

[11] T. Dierks and S. Jagannathan, "Online optimal control of affine nonlinear discrete-time systems with unknown internal dynamics by using time-based policy update," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 7, pp. 1118–1129, July 2012.

[12] D. Zhao, Z. Xia, and D. Wang, "Model-free optimal control for affine nonlinear systems with convergence analysis," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 4, pp. 1461–1468, 2015.

[13] X. Zhong, H. He, H. Zhang, and Z. Wang, "Optimal control for unknown discrete-time nonlinear Markov jump systems using adaptive dynamic programming," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 12, pp. 2141–2155, 2014.

[14] Q. Wei, F. Wang, D. Liu, and X. Yang, "Finite-approximation-error-based discrete-time iterative adaptive dynamic programming," *IEEE Trans. Cybern.*, vol. 44, no. 12, pp. 2820–2833, 2014.

[15] Z. Ni, H. He, D. Zhao, X. Xu, and D. V. Prokhorov, "GrDHP: A general utility function representation for dual heuristic dynamic programming," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 3, pp. 614–627, 2015.

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

[17] F.-Y. Wang, H. Zhang, and D. Liu, "Adaptive dynamic programming: An introduction," *IEEE Comput. Intell. Mag.*, vol. 4, no. 2, pp. 39–47, May 2009.

[18] W. B. Powell, *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. John Wiley & Sons, 2007, vol. 703.

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

[20] M. Abu-Khalaf, F. Lewis, and J. Huang, "Policy iterations on 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.

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

[22] H. Zhang, Q. Wei, and D. Liu, "An iterative adaptive dynamic programming method for solving a class of nonlinear zero-sum differential games," *Automatica*, vol. 47, no. 1, pp. 207–214, 2011.

[23] T. Dierks and S. Jagannathan, "Optimal control of affine nonlinear continuous-time systems using an online Hamilton-Jacobi-Isaacs formulation," in *49th IEEE Conf. Decision and Control (CDC)*, Atlanta, GA, USA, 2010, pp. 3048–3053.

[24] K. G. Vamvoudakis and F. Lewis, "Online solution of nonlinear two-player zero-sum games using synchronous policy iteration," *Int. J. Robust and Nonlinear Control*, vol. 22, no. 13, pp. 1460–1483, 2012.

[25] K. G. Vamvoudakis and F. L. Lewis, "Online actor-critic algorithm to solve the continuous-time infinite horizon optimal control problem," *Automatica*, vol. 46, no. 5, pp. 878–888, 2010.

[26] H.-N. Wu and B. Luo, "Neural network based online simultaneous policy update algorithm for solving the HJI equation in nonlinear H_∞ control," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 12, pp. 1884–1895, Dec 2012.

[27] B. Luo and H.-N. Wu, "Computationally efficient simultaneous policy update algorithm for nonlinear H_∞ state feedback control with Galerkin's method," *Int. J. Robust and Nonlinear Control*, vol. 23, no. 9, pp. 991–1012, 2013.

[28] S. Yasini, A. Karimpour, M.-B. Naghibi Sistani, and H. Modares, "Online concurrent reinforcement learning algorithm to solve two-player zero-sum games for partially unknown nonlinear continuous-time systems," *Int. J. Adaptive Control and Signal Processing*, vol. 29, no. 4, pp. 473–493, 2015.

[29] A. Al-Tamimi, M. Abu-Khalaf, and F. 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.

[30] 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, 2007.

[31] D. Liu, H. Li, and D. Wang, "Neural-network-based zero-sum game for discrete-time nonlinear systems via iterative adaptive dynamic programming algorithm," *Neurocomputing*, vol. 110, pp. 92–100, 2013.

[32] H. Zhang, C. Qin, B. Jiang, and Y. Luo, "Online adaptive policy learning algorithm for H_∞ state feedback control of unknown affine nonlinear discrete-time systems," *IEEE Trans. Cybern.*, vol. 44, no. 12, pp. 2706–2718, Dec 2014.

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

[34] L. Busoniu, R. Babuska, B. D. Schutter, and D. Ernst, *Reinforcement Learning and Dynamic Programming Using Function Approximators*, 1st ed. Boca Raton, FL, USA: CRC Press, Inc., 2010.

[35] D. Zhao and Y. Zhu, "MEC–a near-optimal online reinforcement learning algorithm for continuous deterministic systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 2, pp. 346–356, Feb 2015.

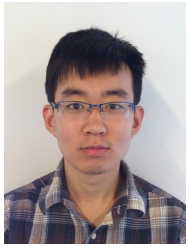
[36] Y. Zhu, D. Zhao, and X. Li, "Using reinforcement learning techniques to solve continuous-time non-linear optimal tracking problem without system dynamics," *IET Control Theory Appl.*, February 2016. [Online]. Available: http://digital-library.theiet.org/content/journals/10.1049/iet-cta.2015.0769

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

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

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

[40] K. G. Vamvoudakis and F. L. Lewis, "Multi-player non-zero-sum games: Online adaptive learning solution of coupled Hamilton-Jacobi equations," *Automatica*, vol. 47, no. 8, pp. 1556–1569, 2011.
[41] H. Zhang, L. Cui, and Y. Luo, "Near-optimal control for nonzero-sum differential games of continuous-time nonlinear systems using single-network ADP," *IEEE Trans. Cybern.*, vol. 43, no. 1, pp. 206–216, Feb 2013.
[42] D. Zhao, Q. Zhang, D. Wang, and Y. Zhu, "Experience replay for optimal control of nonzero-sum game systems with unknown dynamics," *IEEE Trans. Cybern.*, vol. 46, no. 3, pp. 854–865, 2016.
[43] M. Johnson, S. Bhasin, and W. Dixon, "Nonlinear two-player zero-sum game approximate solution using a policy iteration algorithm," in *50th IEEE Conf. Decision and Control and Eur. Control Conf. (CDC-ECC)*, Dec 2011, pp. 142–147.
[44] S. Yasini, M. Sistani, and A. Karimpour, "Approximate dynamic programming for two-player zero-sum game related to H_∞ control of unknown nonlinear continuous-time systems," *Int. J. Control, Autom., Syst.*, vol. 13, no. 1, pp. 99–109, 2015.
[45] H. Modares, F. Lewis, and M.-B. Naghibi-Sistani, "Adaptive optimal control of unknown constrained-input systems using policy iteration and neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 10, pp. 1513–1525, Oct 2013.
[46] D. Liu, H. Li, and D. Wang, "Online synchronous approximate optimal learning algorithm for multi-player non-zero-sum games with unknown dynamics," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 44, no. 8, pp. 1015–1027, Aug 2014.
[47] D. Wang, D. Liu, Q. Wei, D. Zhao, and N. Jin, "Optimal control of unknown nonaffine nonlinear discrete-time systems based on adaptive dynamic programming," *Automatica*, vol. 48, no. 8, pp. 1825–1832, 2012.
[48] Y. Jiang and Z.-P. Jiang, "Robust adaptive dynamic programming for nonlinear control design," in *IEEE 51st Annual Conf. Decision and Control (CDC)*, 2012, pp. 1896–1901.
[49] Z.-P. Jiang and Y. Jiang, "Robust adaptive dynamic programming for linear and nonlinear systems: An overview," *Eur. J. Control*, vol. 19, no. 5, pp. 417–425, 2013.
[50] D. Vrabie and F. Lewis, "Adaptive dynamic programming for online solution of a zero-sum differential game," *J. Control Theory and Applications*, vol. 9, no. 3, pp. 353–360, 2011.
[51] H. Li, D. Liu, and D. Wang, "Integral reinforcement learning for linear continuous-time zero-sum games with completely unknown dynamics," *IEEE Trans. Autom. Sci. Eng.*, vol. 11, no. 3, pp. 706–714, 2014.
[52] H. Modares, F. Lewis, and Z.-P. Jiang, "H_∞ tracking control of completely unknown continuous-time systems via off-policy reinforcement learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 10, pp. 2550–2562, 2015.
[53] L. Kantorovitch, "The method of successive approximation for functional equations," *Acta Mathematica*, vol. 71, no. 1, pp. 63–97, 1939.
[54] R. Tapia, "The Kantorovich theorem for Newton's method," *American Mathematical Monthly*, pp. 389–392, 1971.
[55] B. A. Finlayson, *The Method of Weighted Residuals and Variational Principles*. New York: Academic Press, 1990.
[56] 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.
[57] S. Gratton, A. S. Lawless, and N. K. Nichols, "Approximate Gauss-Newton methods for nonlinear least squares problems," *SIAM J. Optimization*, vol. 18, no. 1, pp. 106–132, 2007.
[58] J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2000.
[59] C. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, no. 3–4, pp. 279–292, 1992.
[60] V. Nevistic and J. A. Primbs, "Constrained nonlinear optimal control: A converse HJB approach," California Institute of Technology, Tech. Rep. 96-021, 1996.
[61] X. Xu, Z. Hou, C. Lian, and H. He, "Online learning control using adaptive critic designs with sparse kernel machines," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 5, pp. 762–775, 2013.
[62] D. Ernst, P. Geurts, and L. Wehenkel, "Tree-based batch mode reinforcement learning," *J. Mach. Learn. Res.*, vol. 6, pp. 503–556, Dec. 2005.



Yuanheng Zhu received the B.S. degree from Nanjing University in 2010 and recieved the Ph.D. degree from Institute of Automation, Chinese Academy of Sciences in 2015.

He is currently an Assistant Professor at the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences. His research interests include optimal control, adaptive dynamic programming, and reinforcement learning.

Dongbin Zhao (M'06-SM'10) received the Ph.D. degrees from Harbin Institute of Technology, Harbin, China, in 2000. Dr. Zhao was a postdoctoral fellow with Tsinghua University, Beijing, China, from May 2000 to Jan. 2002. He was an associate professor from 2002, and now is a professor at the State Key Laboratory of Management and Control for Complex Systems from 2012 with the Institute of Automation, Chinese Academy of Sciences, China. He has published 4 books, and over 50 international journal papers. His current research interests are in the areas of computational intelligence, adaptive dynamic programming, robotics, intelligent transportation systems, and smart grids.

Dr. Zhao is the Associate Editor of IEEE Transactions on Neural Networks and Learning Systems (2012-), IEEE Computation Intelligence Magazine (2014-), etc. He serves as the Chair of Adaptive Dynamic Programming and Reinforcement Learning Technical Committee (2015-), Multimedia Subcommittee (2015-) of IEEE Computational Intelligence Society (CIS). He works as several guest editors of international journals. He is involved in organizing several international conferences.

Xiangjun Li (M'06-SM'12) was born in 1979. He received the B.E. degree in electrical engineering from Shenyang University of Technology, China, in July 2001, and the M.E. and Ph.D. degrees in electrical and electronic engineering from Kitami Institute of Technology (KIT), Japan, in March 2004 and March 2006, respectively. From May 2006 to March 2010, he worked as a postdoctoral research fellow at Korea Institute of Energy Research (KIER), Daejeon, Korea, and Tsinghua University, Beijing, China, respectively. In March 2010, he joined the Electrical Engineering and New Material Department, China Electric Power Research Institute (CEPRI), Beijing, China, where he has been engaged in the topic of integration/control/SCADA/EMS/application technologies for large-scale multi-type battery energy storage system/stations, wind/PV/battery hybrid distributed generation systems, and smart grid. His research interests include renewable energy power generation, electric energy saving/storage technology, internet of things, big data application, real-time power system control, and adaptive dynamic programming. Dr. Li is a senior member of CES and a member of IET.