Received July 26, 2019, accepted August 29, 2019, date of publication September 4, 2019, date of current version September 30, 2019.
Digital Object Identifier 10.1109/ACCESS.2019.2939384

Discrete-Time Multi-Player Games Based on Off-Policy Q-Learning

JINNA LI[1,2], (Member, IEEE), ZHENFEI XIAO[1], AND PING LI[1], (Senior Member, IEEE)
[1]School of Information and Control Engineering, Liaoning Shihua University, Fushun 113001, China
[2]State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, Shenyang 110819, China
Corresponding author: Ping Li (liping@lnpu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61673280, Grant 61673199, Grant 61673100, Grant 61525302, Grant 61590922, and Grant 61503257, in part by the Open Project of Key Field Alliance of Liaoning Provincial Natural Science Foundation under Grant 2019-KF-03-06, in part by the Open Project of State Key Laboratory of Synthetical Automation for Process Industries under Grant PAL-N201603, in part by the Project of Liaoning Province under Grant LR2017006, and in part by the Project of Liaoning Shihua University under Grant 2018XJJ-005.

ABSTRACT In this paper, an off-policy game Q-learning algorithm is proposed for solving linear discrete-time non-zero sum multi-player game problems. Unlike the existing Q-learning methods for solving the Riccati equation by on-policy learning approaches for multi-player games, an off-policy game Q-learning method is developed for achieving the Nash equilibrium of multiple players. To this end, first, a non-zero sum game problem is formulated, and the value function and the Q-function defined according to each-player individual performance index are rigorously proved to be linear quadratic forms. Then, based on the dynamic programming and Q-learning methods, an off-policy game Q-learning algorithm is developed to find the control policies for multi-player games, such that the Nash equilibrium is reached under the learned control policies. The merit of this paper lies in that the proposed algorithm does not require the system model parameters to be known a priori and fully utilizes measurable data to learn the Nash equilibrium solution. Moreover, there is no bias of Nash equilibrium solution when implementing the proposed off-policy game Q-learning algorithm even though probing noises are added to control policies for maintaining the persistent excitation condition. While bias of the Nash equilibrium solution could be produced if on-policy game Q-learning is employed. This is another contribution of this paper.

INDEX TERMS Adaptive dynamic programming, off-policy Q-learning, non-zero sum game, Nash equilibrium, discrete-time systems.

I. INTRODUCTION

Reinforcement learning (RL), as one of machine learning methods, has been widely used in solving optimal control problems [1]–[4] by using partially or completely unknown dynamics for systems with [5]–[12]. The approximate optimal control strategies for varieties of control issues and control systems have been reported in the latest decade, such as [3] for MIMO systems, [5] for multi-agent graphical games, [8], [10], [12] for H_∞ control, [13]–[17] for optimal tracking control, and [18], [19] for Q-learning based controller design, etc.

The on-policy RL and the off-policy RL are two kinds of RL approaches. When conducting the on-policy RL, the data used for learning the optimal control policies are generated by the same control policy as the one under evaluation, while two types of control policies are needed in the off-policy RL. One is the behavior policy used for generating systems data, and the other is the target policy updated until convergence to the optimal control policy. In the literature as mentioned before, the Q-learning algorithms in [18] and [19] are actually the on-policy RL. Other on-policy RL research results can be found in [18]–[22]. Compared with the off-policy learning method, the remarkable shortcomings of the on-policy Rl algorithm [6]–[10] lie in 1) insufficient exploration of the systems; 2) interfering with the operation of systems in the learning process; 3) under the condition of satisfying the persistence of excitation (PE), adding probing noises to the system is proved to produce deviation of solutions to the focused optimization problems.

Q-Learning is a behavior-dependent heuristic dynamic programming, and the research on the off-policy Q-learning

The associate editor coordinating the review of this manuscript and approving it for publication was Bora Onat.

has been attracted increasing attention by scholars. For linear discrete-time (DT) systems, Al-Tamimi *et al.* [21] derived an H_∞ optimal state feedback controller, Kim and Lewis [23] designed the optimal tracking controller and Jiang *et al.* [24] settled the optimal regulation problem. For linear continuous-time (CT) systems, Lee *et al.* [19] and Vamvoudakis [25] focused on the linear quadratic regulation problem. Vamvoudakis [26] engaged in the linear graphical game problem. For nonlinear systems, Luo *et al.* [27] aimed at solving the model-free optimal tracking control problem for affine DT systems and proposed an adaptive optimal controller method of general DT systems [28]. Modares and Lewis [13] focused on the optimal tracking control problem of continuous-time systems. Moreover, an off-policy interleaved Q-learning algorithm was developed for affine nonlinear systems in [29]. It is worth pointing out that the above-mentioned off-policy Q-learning results can be implemented only for designing one single controller that leads systems to the optimum.

Since large scale, complexity and multiple subsystems are the basic features in modern practical industries, then Q-learning of multi-player systems for finding multiple optimal controllers should be investigated by researchers. In non-zero sum or zero sum multi-player games, each player makes efforts to optimize its own performance or reward by learning feedback from the environment and improving its behavior. In [21], the application of model-free Q-learning zero sum game to H_∞ problem has been studied. Vamvoudakis *et al.* [30] systematically summarized game theory-based RL methods to solve two-player games including DT and CT systems. Notice that the off-policy RL algorithm has been proposed in [30] for linear DT multi-player systems, then whether the off-policy Q-learning method can be used to study the optimal control problem of the completely unknown linear DT multi-player games or not? And if it can work, then how to design the off-policy Q-learning algorithm for achieving the Nash equilibrium of linear DT multi-player games using only measured data is the key point. This drops down into our focus. To the authors' best knowledge, the off-policy game Q-learning using only measured data for linear DT multi-player systems with completely unknown model parameters has not been reported up to now.

In this paper, we devote to developing an off-policy game Q-learning algorithm for achieving the Nash equilibrium of linear DT multi-player systems by combining game theory and Q-learning. The contributions of this paper are summarized as follows.

1) Referring to the existed on-policy Q-learning methods [21], [23]–[26], [31] and the off-policy RL method [30] which is for linear DT two-player games, this is the first time to propose an off-policy game Q-learning for solving linear DT multi-player non-zero sum games using only measured data.
2) No bias and bias of Nash equilibrium solution when adding probing noises into multi-player systems are rigorously proved, which are the extension of [8] where the systems with single player or agent are concerned.

The rest of this paper is organized as follows. In Section II, the non-zero sum games problem of linear DT multi-player systems is formulated. Section III devotes to solving the non-zero sum games. In Section IV, an on-policy game Q-learning algorithm is proposed and the bias of the solution is analyzed. In Section V, an off-policy game Q-learning algorithm is proposed, and the rigorous proof of the unbiased solution is presented. The effectiveness of the proposed algorithm is verified by numerical simulations, and the comparisons between the off-policy game Q-learning algorithm and the on-policy game Q-learning algorithm are carried out in Section VI. Section VII states the conclusions in this paper.

Notations: $\mathbb{R}^p$ denotes the p dimensional Euclidean space. $\mathbb{R}^{p\times q}$ is the set of all real p by q matrices. $\otimes$ stands for the Kronecker product. $vec(L)$ is used to turn any matrix L into a single column vector.

II. PROBLEM STATEMENT

In this section, the optimal control problem of linear DT multi-player systems is formulated. Moreover, the value function and the Q-function defined in terms of the cost function of each player are proved to be linear quadratic forms.

Consider the following linear DT multi-player system

$$x_{k+1} = Ax_k + \sum_{i=1}^{n} B_{ik} u_{ik} \tag{1}$$

where $x_k = x(k) \in \mathbb{R}^p$ is the system state, $u_{ik} = u_i(k) \in \mathbb{R}^{m_i}$ $(i = 1, \ldots, n)$ are the control inputs. $A \in \mathbb{R}^{p\times p}$, $B_i \in \mathbb{R}^{p\times m_i}$ and k is the sampling time instant. The full state of system (1) can be accessed by each of the agents or players i. The target of each player is to minimize its own performance index by its efforts, regardless of the performance of other players. The performance index J_i of each player i $(i = 1, 2, \ldots, n)$ is defined as the accumulative sum of utility functions from time instant 0 to infinity as given below [5]:

$$J_i(x_0, u_1, \ldots, u_n) = \sum_{k=0}^{\infty}(x_k^T Q_i x_k + \sum_{q=1}^{n} u_{qk}^T R_q u_{qk}) \tag{2}$$

where Q_i and R_q are respectively positive semi-definite matrices and positive definite matrices. x_0 represents the initial state of system (1) at time instant 0. Minimizing (2) subject to (1) is indeed a standard multi-player non-zero sum games problem, and all players will finally reach the Nash equilibrium. The objective of this article is to find the stabilizable control policies $u_{1k}, u_{2k}, \ldots, u_{nk}$ by using RL combined with game theory, such that the performance index of each player shown in (2) is minimized.

The definition of admissible control policies is given below, which is useful for Assumption 1 and Lemma 1.

Definition 1 [21], [32]: Control policies $u_1(x_k)$,$u_2(x_k)$, $\ldots, u_n(x_k)$ are called the admissible with respect to (2) on $\Omega \in \mathbb{R}^p$, if $u_1(x_k), u_2(x_k), \ldots, u_n(x_k)$ are continuous on Ω, $u_1(0) = 0, u_2(0) = 0, \ldots, u_n(0) = 0,$

$u_1(x_k), u_2(x_k), \ldots, u_n(x_k)$ stabilize (1) on Ω and (2) is finite $\forall x_0 \in \Omega$.

Assumption 1: The n-player system (1) is controllable and there exists at least one set of admissible control policies [32].

According to performance indicator (2), suppose there exists a set of admissible control policies $u_1(x)$, $u_2(x)$, $\ldots$, $u_n(x)$, one can respectively define the following optimal value function and the optimal Q-function for each player i $(i = 1, 2, \ldots, n)$ as [5]:

$$V_i^*(x_k) = \min_{u_i} \sum_{l=k}^{\infty} (x_l^T Q_i x_l + \sum_{q=1}^{n} u_{ql}^T R_q u_{ql}) \tag{3}$$

and

$$Q_i^*(x_k, u_{ik}, u_{-ik}) = x_k^T Q_i x_k + \sum_{q=1}^{n} u_{qk}^T R_q u_{qk} + V_i^*(x_{k+1}) \tag{4}$$

$u_{-ik} = \left[u_{1k}^T, \ldots, u_{i-1,k}^T, \ldots, u_{i+1,k}^T, \ldots, u_{nk}^T\right]^T$. Thus, the following relation holds

$$V_i^*(x_k) = Q_i^*(x_k, u_{ik}^*, u_{-ik}) \quad (i = 1, 2, \ldots, n) \tag{5}$$

Lemma 1: Suppose that the control policies $u_{ik} = -K_i x_k$ and they are admissible, the value function $V_i(x_k)$ and the Q-function $Q_i(x_k, u_{ik}, u_{-ik})$ of each player i can be respectively expressed as the following quadratic forms.

$$V_i(x_k) = x_k^T P_i x_k \tag{6}$$

and

$$Q_i(x_k, u_{ik}, u_{-ik}) = z_k^T H_i z_k \tag{7}$$

where P_i and H_i are positive definite matrices. And

$$z_k = \begin{bmatrix} x_k^T & u_1^T & u_2^T & \ldots & u_n^T \end{bmatrix}^T \tag{8}$$

Proof:

$$\begin{aligned} V_i(x_k) &= \sum_{l=k}^{\infty} (x_l^T Q_i x_l + \sum_{q=1}^{n} u_{ql}^T R_q u_{ql}) \\ &= \sum_{l=k}^{\infty} \left[x_l^T Q_i x_l + \sum_{q=1}^{n} (-K_q x_l)^T R_q (-K_q x_l) \right] \\ &= \sum_{l=0}^{\infty} x_{l+k}^T \left[Q_i + \sum_{q=1}^{n} (K_q)^T R_q (K_q) \right] x_{l+k} \end{aligned} \tag{9}$$

where $x_{l+k} = (A - \sum_{i=1}^{n} B_i K_i)^l x_k = G^l x_k$. Further, one has

$$V_i(x_k) = \sum_{l=0}^{\infty} x_k^T (G^l)^T \left[Q_i + \sum_{q=1}^{n} (K_q)^T R_q (K_q) \right] (G^l) x_k \tag{10}$$

then, one has

$$V_i(x_k) = x_k^T P_i x_k \tag{11}$$

where

$$P_i = \sum_{l=0}^{\infty} (G^l)^T \left[Q_i + \sum_{q=1}^{n} (K_q)^T R_q (K_q) \right] (G^l)$$

Then, one has

$$\begin{aligned} &Q_i(x_k, u_{ik}, u_{-ik}) \\ &\quad = x_k^T Q_i x_k + \sum_{q=1}^{n} u_{qk}^T R_q u_{qk} + V_i^*(x_{k+1}) \\ &\quad = x_k^T Q_i x_k + \sum_{q=1}^{n} u_{qk}^T R_q u_{qk} + (A x_k + \sum_{i=1}^{n} B_i u_i)^T P_i \\ &\qquad \times (A x_k + \sum_{i=1}^{n} B_i u_i) \\ &\quad = z_k^T H_i z_k \end{aligned} \tag{12}$$

where

$$H_i = \begin{bmatrix} H_{i,xx} & H_{i,xu_1} & H_{i,xu_2} & \ldots & H_{i,xu_n} \\ H_{i,xu_1}^T & H_{i,u_1u_1} & H_{i,u_1u_2} & \ldots & H_{i,u_1u_n} \\ H_{i,xu_2}^T & H_{i,u_1u_2}^T & H_{i,u_2u_2} & \ldots & H_{i,u_2u_n} \\ \vdots & \vdots & \vdots & \ldots & \vdots \\ H_{i,xu_n}^T & H_{i,u_1u_n}^T & H_{i,u_2u_n}^T & \ldots & H_{i,u_nu_n} \end{bmatrix}$$
$$= \begin{bmatrix} A^T P_i A + Q_i & A^T P_i B_1 & \ldots & A^T P_i B_n \\ (A^T P_i B_1)^T & B_1^T P_i B_1 + R_1 & \ldots & B_1^T P_i B_n \\ (A^T P_i B_2)^T & (B_1^T P_i B_2)^T & \ldots & B_2^T P_i B_n \\ \vdots & \vdots & \ldots & \vdots \\ (A^T P_i B_n)^T & (B_1^T P_i B_n)^T & \ldots & B_n^T P_i B_n + R_n \end{bmatrix} \tag{13}$$

By (11) and (12) , one can get

$$P_i = M^T H_i M \tag{14}$$

where

$$M = \begin{bmatrix} I & -K_1^T & -K_2^T & \ldots & -K_n^T \end{bmatrix}^T$$

■

III. SOLVING NON-ZERO SUM GAME PROBLEMS

This section deals with solving the non-zero sum games problem.

In the non-zero sum games, it is desired for all players to reach the Nash equilibrium by assuming that each player has the same hierarchical level as others. The definition of Nash equilibrium is given as following.

Definition 2 [30]: If there exists an n-tuple of control strategies $(u_1^*, u_2^*, \ldots, u_n^*)$ satisfying the following n inequalities

$$\begin{aligned} J_i^* &\equiv J_i(u_1^*, u_2^*, \ldots, u_i^*, \ldots, u_n^*) \\ &\leq J_i(u_1^*, u_2^*, \ldots, u_i, \ldots, u_n^*) (i = 1, 2, \ldots, n) \end{aligned}$$

then, this n-tuple of control strategies constitutes the Nash equilibrium solution of n-player finite game (1). And the

n-tuple of quantities $(J_1^*, \dots, J_n^*)$ is the Nash equilibrium outcome of n-player games (1) with respect to (2).

Now, we are in the position of solving the non-zero sum game to find the Nash equilibrium solution. According to the dynamic programming, the following Q-function based game Bellman equations can be derived based on Lemma 1.

$$z_k^T H_i z_k = x_k^T Q_i x_k + \sum_{q=1}^{n} u_{qk}^T R_q u_{qk} + z_{k+1}^T H_i z_{k+1} \quad (15)$$

The optimal control policy u_{ik}^* of each player i should satisfy $\frac{\partial Q_i^*(x_k, u_{ik}, u_{-ik})}{\partial u_{ik}} = 0$ in terms of the necessary condition of optimality. Therefore, one has

$$u_i^*(k) = -K_i^* x_k \quad (16)$$

where

$$K_i^* = H_{i,u_iu_i}^{-1} \Big[H_{i,xu_i}^T - (H_{i,u_iu_1} K_1 + \dots + H_{i,u_iu_{i-1}} K_{i-1} + H_{i,u_iu_{i+1}} K_{i+1} + \dots + H_{i,u_iu_n} K_n) \Big] \quad (17)$$

Substituting K_i^* in (17) into (15) yields the optimal Q-function based Riccati equations.

$$(z_k)^T H_i^* z_k = x_k^T Q_i x_k + \sum_{q=1}^{n} (u_{qk}^*)^T R_q u_{qk}^* + (z_{k+1})^T H_i^* z_{k+1} \quad (18)$$

Where $u_{qk}^* = -K_q^* x_k$. Note that Vamvoudakis et al [30] has proven that the following K_i^* $(i = 1, 2, \dots n)$ guarantee system (1) to be stable and achieved Nash equilibrium of all players can be.

$$K_i^* = (H_{i,u_iu_i}^*)^{-1} \Big[(H_{i,xu_i}^*)^T - (H_{i,u_iu_1}^* K_1^* + \dots + H_{i,u_iu_{i-1}}^* K_{i-1}^* + H_{i,u_iu_{i+1}}^* K_{i+1}^* + \dots + H_{i,u_iu_n}^* K_n^*) \Big] \quad (19)$$

Remark 1: From (17) and (19), it can be seen that in Riccati equations (18) the matrices H_i^* are coupled with each other, and the values of matrices K_i^* are also coupled with each other, which make it difficult to solve Riccati equations (18). Therefore, the on-policy game Q-learning algorithm and off-policy game Q-learning algorithm are given in Section IV and Section V to learn the optimal control laws $u_i^*(k) = -K_i^* x_k$.

IV. FINDING K_i^* $(i = 1, 2, \dots, n)$ BY THE ON-POLICY APPROACH

In this section, the model-free on-policy game Q-learning algorithm is presented and the bias of solution to iterative Q-function based Bellman equations is proved. By learning the Q-function matrices H_i^* in (18), the approximately optimal controller gains of multiple players can be obtained.

A. ON-POLICY GAME Q-LEARNING ALGORITHM

The on-policy RL methods in [18]–[22] are extended to the case of multi-player systems, thus we present on-policy game Q-learning algorithm.

Algorithm 1 On-Policy Game Q-Learning

```
1: Initialization: Given the admissible controller gains for $n$ players $K_1^0, K_2^0, \dots, K_n^0$. Let $j = 0$ and $i = 1$, where $j$ denotes the iteration index and $i$ stands for player $i$ $(i = 1, 2, \dots, n)$;
2: Policies evaluation: solve the Q-function matrices $H_i^{j+1}$
   $z_k^T H_i^{j+1} z_k = x_k^T Q_i x_k + \sum_{q=1}^{n} (u_{qk}^j)^T R_q u_{qk}^j + z_{k+1}^T H_i^{j+1} z_{k+1}$   (20)
3: Policies update:
   $u_{ik}^{j+1} = -K_i^{j+1} x_k$   (21)
   where
   $K_i^{j+1} = (H_{i,u_iu_i}^{j+1})^{-1} \Big[ (H_{i,xu_i}^{j+1})^T - (H_{i,u_iu_1}^{j+1} K_1^j + \dots + H_{i,u_iu_{i-1}}^{j+1} K_{i-1}^j + H_{i,u_iu_{i+1}}^{j+1} K_{i+1}^j + \dots + H_{i,u_iu_n}^{j+1} K_n^j) \Big]$   (22)
4: If $i < n$, then $i = i + 1$ and go back to Step 2. Otherwise $j = j + 1$, $i = 1$, and go to Step 5;
5: Stop when
   $\left\| H_i^{j-1} - H_i^j \right\| \le \varepsilon \quad (i = 1, 2, \dots, n)$
   with a small constant $\varepsilon$ $(\varepsilon > 0)$. Otherwise go back to Step 2.
```

Remark 2: Algorithm 1 calculating Q-function matrices H_i^{j+1} yields the updated K_i^{j+1} $(H_i^{j+1} \to K_i^{j+1})$. As $j \to \infty$, H_i^{j+1} converge to H_i^* result in the convergence of K_i^{j+1} to K_i^* for all players, which can be proved in the similar way to [29], [30], [33].

B. BIAS ANALYSIS OF SOLUTION FOR THE ON-POLICY GAME Q-LEARNING ALGORITHM

To satisfy the PE condition in Algorithm 1, probing noises are added to system (1). Thus, the actual control inputs applied to the system for collecting data are

$$\hat{u}_{ik}^j = u_{ik}^j + e_{ik} \quad (23)$$

with $e_{ik} = e_i(k)$ being probing noises and u_{ik}^j given by (21). Theorem 1 will prove the bias of solution to (20).

Theorem 1: Rewrite Bellman equation (20) as

$$x_k^T (M^j)^T H_i^{j+1} M^j x_k = x_k^T Q_i x_k + \sum_{q=1}^{n} (u_{qk}^j)^T R_q u_{qk}^j + x_{k+1}^T (M^j)^T H_i^{j+1} M^j x_{k+1} \quad (24)$$

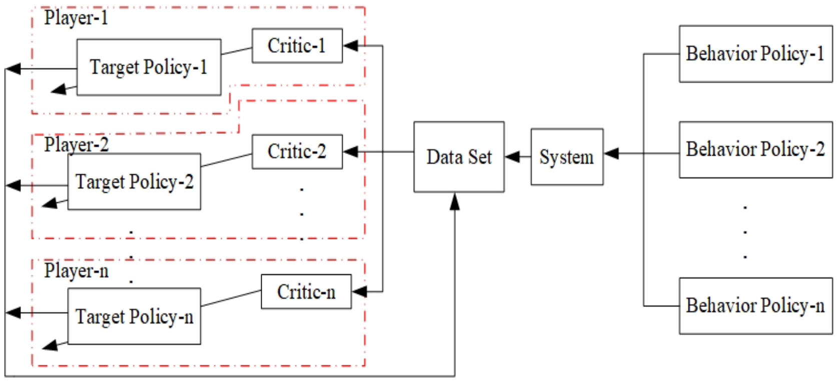

FIGURE 1. Architecture of the off-policy game Q-learning.

where

$$M^j = \begin{bmatrix} I & -(K_1^j)^T & -(K_2^j)^T & \dots & -(K_n^j)^T \end{bmatrix}^T$$

Let H_i^{j+1} be the solution (24) with $e_{ik} = 0$ and $\hat{H}_i^{j+1}$ be the solution to (24) with $e_{ik} \neq 0$. Then, $H_i^{j+1} \neq \hat{H}_i^{j+1}$.

Proof: Using (23) with $e_{ik} \neq 0$ in (24), Bellman equation (24) becomes the following

$$\begin{aligned} x_k^T (M^j)^T \hat{H}_i^{j+1} M^j x_k &= x_k^T Q_i x_k + \sum_{q=1}^{n} (u_{qk}^j + e_{qk})^T R_q \\ &\quad \times (u_{qk}^j + e_{qk}) + x_{k+1}^T (M^j)^T \hat{H}_i^{j+1} M^j x_{k+1} \end{aligned} \tag{25}$$

where

$$x_{k+1} = A x_k + \sum_{i=1}^{n} B_i (u_{ik}^j + e_{ik}) \tag{26}$$

Further, (25) is rewritten as

$$\begin{aligned} x_k^T (M^j)^T \hat{H}_i^{j+1} M^j x_k &= x_k^T Q_i x_k \\ &\quad + \sum_{q=1}^{n} (u_{qk}^j + e_{qk})^T R_q (u_{qk}^j + e_{qk}) \\ &\quad + (A x_k + \sum_{i=1}^{n} B_i (u_{ik}^j + e_{ik}))^T (M^j)^T \hat{H}_i^{j+1} \\ &\quad \times M^j (A x_k + \sum_{i=1}^{n} B_i (u_{ik}^j + e_{ik})) \\ &= x_k^T Q_i x_k + \sum_{q=1}^{n} (u_{qk}^j)^T R_q u_{qk}^j \\ &\quad + x_{k+1}^T (M^j)^T \hat{H}_i^{j+1} M^j x_{k+1} + 2 \sum_{q=1}^{n} e_{qk}^T R_q u_{qk}^j \\ &\quad + \sum_{i=1}^{n} e_{ik}^T (B_i^T (M^j)^T \hat{H}_i^{j+1} M^j B_i + R_i) e_{ik} \\ &\quad + 2 \sum_{i=1}^{n} e_{ik}^T B_i^T (M^j)^T \hat{H}_i^{j+1} M^j x_{k+1} \end{aligned} \tag{27}$$

It can be concluded that the solution to (27) is not the solution to (24) if $e_{ik} \neq 0$. Since the solution to (24) is equivalent to the solution to (20), then one has $H_i^{j+1} \neq \hat{H}_i^{j+1}$. Hence, adding probing noises during implementing the proposed on-policy game Q-learning Algorithm 1 can produce bias of solution. This completes the proof. ■

Remark 3: It is worth noting that data are generated by $u_i^{j+1}(k) = -K_i^{j+1} x_k$ in Algorithm 1, which is the typical characteristic of the on-policy approach. Theorem 1 proves that the solution of the on-policy game Q-learning algorithm is biased.

Remark 4: In contrast to [8], we extend the proof of biased solution to iterative Bellman equations in [8] to the case of multi-player non-zero sum games.

V. FINDING K_i^* ($i = 1, 2, \dots, n$) BY THE OFF-POLICY APPROACH

In this section, an off-policy game Q-learning method is proposed to solve the non-zero sum game problem of multiple players. Moreover, the unbiasedness of solution to Q-learning based iterative game Bellman equation is rigorously proved even though adding probing noises ensures the PE condition.

A. DERIVATION OF OFF-POLICY GAME Q-LEARNING ALGORITHM

From (20), one has

$$\begin{aligned} (M^j)^T H_i^{j+1} M^j &= (M^j)^T \Lambda_i M^j + \left(A - \sum_{i=1}^{n} B_i K_i^j \right)^T \\ &\quad \times (M^j)^T H_i^{j+1} M^j \left(A - \sum_{i=1}^{n} B_i K_i^j \right) \end{aligned} \tag{28}$$

where

$$\Lambda_i = diag(Q_i, R_1, R_2, \dots, R_n)$$

Adding auxiliary variables $u_{ik}^j = -K_i^j x_k$ $(i = 1, 2, \dots, n)$ to system (1) yields

$$x_{k+1} = A_c x_k + \sum_{i=1}^{n} B_i (u_{ik} - u_{ik}^j) \tag{29}$$

where $A_c = A - \sum_{i=1}^{n} B_i K_i^j$, u_{ik} are called the behavior control policies to generate data and u_{ik}^j are called the target control policies that players need to learn. When the system trajectory is (29), one has

$$\begin{aligned}
&Q_i^{j+1}(x_k, u_{ik}, u_{-ik}) - x_k^T A_c^T (M^j)^T H_i^{j+1} M^j A_c x_k \\
&\quad = x_k^T (M^j)^T H_i^{j+1} M^j x_k - \left(x_{k+1} - \sum_{i=1}^{n} B_i(u_{ik} - u_{ik}^j)\right)^T \\
&\qquad \times (M^j)^T H_i^{j+1} M^j \left(x_{k+1} - \sum_{i=1}^{n} B_i(u_{ik} - u_{ik}^j)\right) \\
&\quad = x_k^T (M^j)^T \Lambda_i M^j x_k
\end{aligned} \tag{30}$$

In view that P_i^{j+1} and H_i^{j+1} are related as shown in (13) and (14), then the following holds

$$\begin{aligned}
&x_k^T (M^j)^T H_i^{j+1} M^j x_k - x_{k+1}^T (M^j)^T H_i^{j+1} M^j x_{k+1} \\
&\quad + 2\left(A x_k + \sum_{i=1}^{n} B_i u_{ik}\right)^T P_i^{j+1} \sum_{i=1}^{n} B_i(u_{ik} - u_{ik}^j) \\
&\quad - \sum_{i=1}^{n} (u_{ik} - u_{ik}^j)^T B_i^T P_i^{j+1} \sum_{i=1}^{n} B_i(u_{ik} - u_{ik}^j) \\
&\quad = x_k^T (M^j)^T \Lambda_i M^j x_k
\end{aligned} \tag{31}$$

Further one has

$$\begin{aligned}
&x_k^T (M^j)^T H_i^{j+1} M^j x_k - x_{k+1}^T (M^j)^T H_i^{j+1} M^j x_{k+1} \\
&\quad + 2x_k^T \left[H_{i,xu_1}^{j+1} \quad H_{i,xu_2}^{j+1} \ldots H_{i,xu_n}^{j+1}\right] \sum_{i=1}^{n} (u_{ik} + k_i^j x_k) \\
&\quad + 2\sum_{i=1}^{n} u_{ik}^T G_i^{j+1} \sum_{i=1}^{n} (u_{ik} + k_i^j x_k) \\
&\quad - \sum_{i=1}^{n} \left(u_{ik} + k_i^j x_k\right)^T G_i^{j+1} \sum_{i=1}^{n} (u_{ik} + k_i^j x_k) \\
&\quad = x_k^T (M^j)^T \Lambda_i M^j x_k
\end{aligned} \tag{32}$$

where

$$G_i^{j+1} = \begin{bmatrix}
H_{i,u_1u_1}^{j+1} - R_1 & H_{i,u_1u_2}^{j+1} & \ldots & H_{i,u_1u_n}^{j+1} \\
(H_{i,u_1u_2}^{j+1})^T & H_{i,u_2u_2}^{j+1} - R_2 & \ldots & H_{i,u_2u_n}^{j+1} \\
(H_{i,u_1u_3}^{j+1})^T & (H_{i,u_2u_3}^{j+1})^T & \ldots & H_{i,u_3u_n}^{j+1} \\
\vdots & \vdots & \ldots & \vdots \\
(H_{i,u_1u_n}^{j+1})^T & (H_{i,u_2u_n}^{j+1})^T & \ldots & H_{i,u_nu_n}^{j+1} - R_n
\end{bmatrix}$$

Manipulating (32) can get the following form

$$\hat{\theta}_i^j(k) \hat{L}_i^{j+1} = \hat{\rho}_{i,k} \tag{33}$$

where

$$\begin{aligned}
\hat{\rho}_{i,k} &= x_k^T Q_i x_k + \sum_{i=1}^{n} u_{ik}^T R_i u_{ik} \\
\hat{L}_i^{j+1} &= \left[(vec(\hat{L}_{i,rz}^{j+1}))^T, \ldots, (vec(\hat{L}_{i,nn}^{j+1}))^T\right]^T \\
\hat{\theta}_i^j(k) &= \left[\hat{\theta}_{i,rz}^j, \ldots, \hat{\theta}_{i,nn}^j\right]
\end{aligned}$$

with $r = 0, 1, 2, \ldots, n$, $z = r, r+1, r+2, \ldots, n$.

$$\begin{aligned}
\hat{\theta}_{i,00}^j &= x_k^T \otimes x_k^T - x_{k+1}^T \otimes x_{k+1}^T \\
\hat{L}_{i,00}^{j+1} &= H_{i,xx}^{j+1} \\
\hat{\theta}_{i,ss}^j &= -(K_s^j x_{k+1})^T \otimes (K_s^j x_{k+1})^T + u_s^T \otimes u_s^T \\
\hat{L}_{i,ss}^{j+1} &= H_{i,u_su_s}^{j+1} \\
\hat{\theta}_{i,0s}^j &= 2x_{k+1}^T \otimes (K_s^j x_{k+1})^T + 2x_k^T \otimes u_s^T \\
\hat{L}_{i,0s}^{j+1} &= H_{i,xu_s}^{j+1} \\
\hat{\theta}_{i,st}^j &= -2(K_s^j x_{k+1})^T \otimes (K_t^j x_{k+1})^T + 2u_s^T \otimes u_t^T \\
\hat{L}_{i,st}^{j+1} &= H_{i,u_su_t}^{j+1}
\end{aligned}$$

with $s \neq t$ and $s, t = 1, 2, \ldots, n$.

Based on the above part, $K_1^{j+1}, K_2^{j+1}, \ldots, K_n^{j+1}$ can be expressed as the form of $\hat{L}_i^{j+1}$

$$\begin{aligned}
K_i^{j+1} = (\hat{L}_{i,ii}^{j+1})^{-1} \Big((\hat{L}_{i,0i}^{j+1})^T - \Big[(\hat{L}_{i,i1}^{j+1})^T K_1^j + \ldots \\
+ (\hat{L}_{i,(i,i-1)}^{j+1})^T K_{i-1}^j + \hat{L}_{i,(i,i+1)}^{j+1})^T K_{i+1}^j \\
+ \cdots + (\hat{L}_{i,in}^{j+1})^T K_n^j\Big]\Big)
\end{aligned} \tag{34}$$

Theorem 2: (H_i^{j+1}, K_i^{j+1}) are the solution of (33) and (34) if and only if they are the solution of (20) and (22).

Proof: From the derivation of (33) and (34), one can conclude that if (H_i^{j+1}, K_i^{j+1}) are the solution of (20) and (22), then (H_i^{j+1}, K_i^{j+1}) satisfies (33) and (34). Next, we will prove that the solution to (33) and (34) is also the solution to (20) and (22).

It is obvious that (33) is equivalent to (31), so the solutions of both (33) and (31) are going to be the same. If (H_i^{j+1}, K_i^{j+1}) is the solution to (31), then it is also the solution to (30) based on Lemma 1. Subtracting (31) from (30) yields (20), then the solution to (31) is the same one to (20). Thus (H_i^{j+1}, K_i^{j+1}) satisfied with (33) makes (20) hold resulting in (34) being (22). ■

Remark 5: If $\hat{L}_i^{j+1}$ are solved correctly, then the $u_{ik}^j = -K_i^j x_k$ can be learned by Algorithm 2. When $j \to \infty$, $u_{ik}^j \to u_{ik}^*$. Since the solution to Algorithm 2 is the same as the solution to Algorithm 1, and u_{ik}^j learned by Algorithm 1 have been proven to converge to u_{ik}^*, then u_{ik}^j found by using Algorithm 2 can converge to u_{ik}^*, under which the Nash equilibrium of multi-player games can be reached.

Remark 6: Algorithm 2 is indeed an off-policy Q-learning approach, since the target control policies are updated but they are not applied to the learning process. The use of arbitrary admissible behavior control policies u_{ik} to generate data and enrich data exploration is the essential feature of the off-policy learning as opposed to the on-policy learning [4], [23]–[26].

Algorithm 2 Off-Policy Game Q-Learning

```
1: Data collection: Collect data $x_k$ by using behavior control policies $u_{ik}$ $(i = 1, 2, \ldots, n)$;
2: Initialization: Given initial admissible controller gains of multiple players $K_1^0, K_2^0, K_3^0, \ldots, K_n^0$. Let $j = 1$ and $i = 1$, where $j$ denotes the iteration index and $i$ stands of player $i$;
3: Implementing the off-policy game Q-learning: By using recursive least-square methods, $\hat{L}_i^{j+1}$ can be calculated using (33). And then $K_i^{j+1}$ is updated by (34);
4: If $i < n$, then $i = i + 1$ and go back to Step 3. Otherwise $j = j + 1$, $i = 1$ and go to Step 5;
5: Stop when $\left\| K_i^j - K_i^{j-1} \right\| \leq \varepsilon$ $(i = 1, 2, \ldots, n)$, the optimal control policy is obtained. Otherwise, $i = 1$, and go back to Step 3.
```

B. NO BIAS ANALYSIS OF SOLUTION FOR THE OFF-POLICY Q-LEARNING ALGORITHM

To satisfy the condition of PE, probing noises are added into (20) in Algorithm 1. It has been proven that Algorithm 1 could produce bias of solution if adding probing noises into systems. The following will prove that the proposed Algorithm 2 does not produce deviation of the solution under the circumstance of adding probing noises.

Theorem 3: Add probing noises to the behavior control policies in Algorithm 2. Let H_i^{j+1} be the solution to (30) with $e_{ik} = 0$ and $\hat{H}_i^{j+1}$ be the solution to (30) with $e_{ik} \neq 0$, then $\hat{H}_i^{j+1} = H_i^{j+1}$.

Proof: After probing noises are added to the behavior control policies, that is $u_{ik} + e_{ik}$, solving (30) is equivalent to solving the following form

$$\hat{x}_k^T (M^j)^T \hat{H}_i^{j+1} M^j \hat{x}_k = \hat{x}_k^T (M^j)^T \Lambda_i M^j \hat{x}_k + \hat{x}_k^T \left(A - \sum_{i=1}^{n} B_i K_i^j\right)^T (M^j)^T \hat{H}_i^{j+1} \times M^j \left(A - \sum_{i=1}^{n} B_i K_i^j\right) \hat{x}_k \tag{35}$$

Notice that if adding probing noises into system (29), then it becomes

$$\hat{x}_{k+1} = A_c \hat{x}_k + \sum_{i=1}^{n} B_i (u_{ik} + e_{ik} + K_i^j \hat{x}_k) \tag{36}$$

In this case, (30) becomes

$$\hat{x}_k^T (M^j)^T \hat{H}_i^{j+1} M^j \hat{x}_k - \left(\hat{x}_{k+1} - \sum_{i=1}^{n} B_i (u_{ik} + e_{ik} + K_i^j \hat{x}_k)\right)^T (M^j)^T \hat{H}_i^{j+1} M^j \times \left(\hat{x}_{k+1} - \sum_{i=1}^{n} B_i (u_{ik} + e_{ik} + K_i^j \hat{x}_k)\right) = \hat{x}_k^T (M^j)^T \Lambda_i M^j \hat{x}_k \tag{37}$$

Substituting (36) into (37), (37) becomes (35). So the solution to (35) is the same as (30). From the proof of Theorem 2, one can find that the solution to (33) is equal to that to (30). Therefore, it is impossible for the off-policy game Q-learning algorithm to produce deviations when adding probing noises. The unbiasedness of the solution of the off-policy game Q-learning is proved. ■

Remark 7: Different from [8], where the unbiased proof of the solution of the off-policy RL algorithm for solving the zero sum game problem with two players was developed, while here the unbiased proof of the solution of the off-policy game Q-learning algorithm is for the non-zero sum multi-player games.

Remark 8: Compared with the off-policy RL algorithm for zero sum games [30], the off-policy game Q-learning algorithm is put forward for the first time for non-zero sum multi-players games and the rigorous proof of unbiased solution even though adding probing noises to keep PE condition is presented in this paper, which has not been reported up to now.

VI. SIMULATION RESULTS

In this section, the effectiveness of the proposed off-policy game Q-learning algorithm is verified respectively for the three-player games and the five-player games.

A. SIMULATION RESULTS FOR THE THREE-PLAYER SYSTEM

Consider the following linear DT system with three players, which play non-zero sum game.

$$x_{k+1} = Ax_k + B_1 u_1 + B_2 u_2 + B_3 u_3 \tag{38}$$

where

$$A = \begin{bmatrix} 0.906488 & 0.0816012 & -0.0005 \\ 0.074349 & 0.90121 & -0.000708383 \\ 0 & 0 & 0.132655 \end{bmatrix}$$

$$B_1 = \begin{bmatrix} -0.00150808 \\ -0.0096 \\ 0.867345 \end{bmatrix}, \quad B_2 = \begin{bmatrix} 0.00951892 \\ 0.00038373 \\ 0 \end{bmatrix}$$

$$B_3 = \begin{bmatrix} -0.00563451 \\ -0.08962 \\ 0.356478 \end{bmatrix}$$

Choose $Q_1 = diag(4, 4, 4)$, $Q_2 = diag(5, 5, 5)$ $Q_3 = diag(6, 6, 6)$ and $R_1 = R_2 = R_3 = 1$. Rewrite (18) as

$$H_i^* = \Lambda_i + \begin{bmatrix} A & B_1 & B_2 & B_3 \\ K_1 A & K_1 B_1 & K_1 B_2 & K_1 B_3 \\ K_2 A & K_2 B_1 & K_2 B_2 & K_2 B_3 \\ K_3 A & K_3 B_1 & K_3 B_2 & K_3 B_3 \end{bmatrix}^T H_i^* \times \begin{bmatrix} A & B_1 & B_2 & B_3 \\ K_1 A & K_1 B_1 & K_1 B_2 & K_1 B_3 \\ K_2 A & K_2 B_1 & K_2 B_2 & K_2 B_3 \\ K_3 A & K_3 B_1 & K_3 B_2 & K_3 B_3 \end{bmatrix} \tag{39}$$

TABLE 1. Optimal controller gains under three probing noises.

	on-policy Q-learning	off-policy Q-learning
Case 1	$K_1=[-0.2671\quad -0.4377\quad -0.0992]$ $K_2=[-0.2679\quad -0.1124\quad -0.0005]$ $K_3=[0.9416\quad 1.5184\quad -0.0504]$	$K_1=[-0.2671\quad -0.4385\quad -0.0992]$ $K_2=[-0.2678\quad -0.1127\quad -0.0006]$ $K_3=[0.9431\quad 1.5190\quad -0.0504]$
Case 2	$K_1=[-0.2927\quad -0.5853\quad -0.1019]$ $K_2=[-0.4230\quad -0.5780\quad -0.0241]$ $K_3=[1.0261\quad 1.9871\quad -0.0415]$	$K_1=[-0.2671\quad -0.4385\quad -0.0992]$ $K_2=[-0.2678\quad -0.1127\quad -0.0006]$ $K_3=[0.9431\quad 1.5190\quad -0.0504]$
Case 3	$K_1=[-0.4736\quad 0.0436\quad -0.0942]$ $K_2=[-0.6749\quad 3.7554\quad 0.0575]$ $K_3=[1.6008\quad -0.0117\quad -0.0660]$	$K_1=[-0.2671\quad -0.4385\quad -0.0992]$ $K_2=[-0.2678\quad -0.1127\quad -0.0006]$ $K_3=[0.9431\quad 1.5190\quad -0.0504]$

The model-based iterative algorithm in terms of (39) is used in MATLAB to obtain the real solution. Thus, the optimal Q-function matrices (H_1^*, H_2^*, H_3^*) and the optimal controller gains (K_1^*, K_2^*, K_3^*) can be obtained.

$$H_1^* = \begin{bmatrix} 26.6577 & 11.7777 & -0.0150 & -0.1100 & 0.2336 & -1.0988 \\ 11.7777 & 21.4849 & -0.0111 & -0.1434 & 0.1164 & -1.6897 \\ -0.0150 & -0.0111 & 4.0707 & 0.4622 & -0.0002 & 0.1910 \\ -0.1100 & -0.1434 & 0.4622 & 4.0223 & -0.0011 & 1.2555 \\ 0.2336 & 0.1164 & -0.0002 & -0.0011 & 1.0024 & -0.0108 \\ -1.0988 & -1.6897 & 0.1910 & 1.2555 & -0.0108 & 1.6737 \end{bmatrix}$$

$$H_2^* = \begin{bmatrix} 32.0935 & 13.2638 & -0.0172 & -0.1220 & 0.2797 & -1.2318 \\ 13.2638 & 25.0310 & -0.0117 & -0.1585 & 0.1310 & -1.9340 \\ -0.0172 & -0.0117 & 5.0883 & 0.5773 & -0.0002 & 0.2383 \\ -0.1220 & -0.1585 & 0.5773 & 4.7753 & -0.0012 & 1.5664 \\ 0.2797 & 0.1310 & -0.0002 & -0.0012 & 1.0029 & -0.0121 \\ -1.2318 & -1.9340 & 0.2383 & 1.5664 & -0.0121 & 1.8244 \end{bmatrix}$$

$$H_3^* = \begin{bmatrix} 37.5293 & 14.7499 & -0.0194 & -0.1340 & 0.3258 & -1.3649 \\ 14.7499 & 28.5771 & -0.0124 & -0.1736 & 0.1455 & -2.1784 \\ -0.0194 & -0.0124 & 6.1059 & 0.6925 & -0.0002 & 0.2857 \\ -0.1340 & -0.1736 & 0.6925 & 5.5283 & -0.0013 & 1.8773 \\ 0.3258 & 0.1455 & -0.0002 & -0.0013 & 1.0034 & -0.0134 \\ -1.3649 & -2.1784 & 0.2857 & 1.8773 & -0.0134 & 1.9750 \end{bmatrix}$$

$$K_1^* = \begin{bmatrix} -0.2671 & -0.4385 & -0.0992 \end{bmatrix}$$
$$K_2^* = \begin{bmatrix} -0.2678 & -0.1127 & -0.0006 \end{bmatrix}$$
$$K_3^* = \begin{bmatrix} 0.9431 & 1.5190 & -0.0504 \end{bmatrix} \tag{40}$$

Three different probing noises were added to verify the unbiasedness of the off-policy game Q-learning algorithm, and compare it with the on-policy game Q-learning algorithm. For ensure PE conditions when solving (20) and (33) used the following three cases of probing noises are

1) Case 1:

$$e_i = \sum_{j}^{100} 0.5 * sin(noise_{feq}(1,j) * k) \tag{41}$$

2) Case 2:

$$e_i = \sum_{j}^{100} 6 * sin(noise_{feq}(1,j) * k) \tag{42}$$

3) Case 3:

$$e_i = \sum_{j}^{100} 10 * sin(noise_{feq}(1,j) * k) \tag{43}$$

where

$$noise_{feq}(1,j) = 500 * rand(1, 100) - 200 * ones(1, 100) \tag{44}$$

Table 1 shows the optimal controller gains when implementing Algorithm 1 and Algorithm 2 under the three cases of probing noises. It can be observed that the solution when using on-policy game Q-learning Algorithm 1 is affected by the probing noises interference and the learned controller gains are biased. On the contrary, for all of the three probing noises, the Q-function matrices H_i^j and controller gains K_i^j always converge to the optimal values when implementing off-policy game Q-learning Algorithm 2.

Simulation results when utilizing the on-policy game Q-learning Algorithm 1: Under the probing noises Case 1, Fig. 2 and Fig. 3 show the errors between the matrices H_i and H_i^* and the errors between the controller gains K_i and K_i^*. Fig. 4 shows the state responses of the game system. The optimal performance J_i is plotted using the learned optimal control policy in Fig. 5.

Fig. 6 and Fig. 7 show the errors between the matrices H_i and H_i^* and the errors between the controller gains K_i and

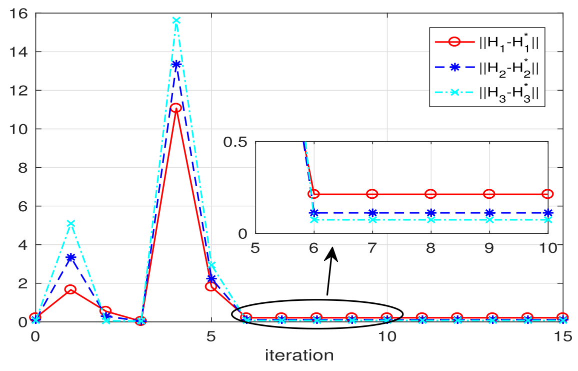

FIGURE 2. Case 1: Convergence of H_i in on-policy game Q-learning.

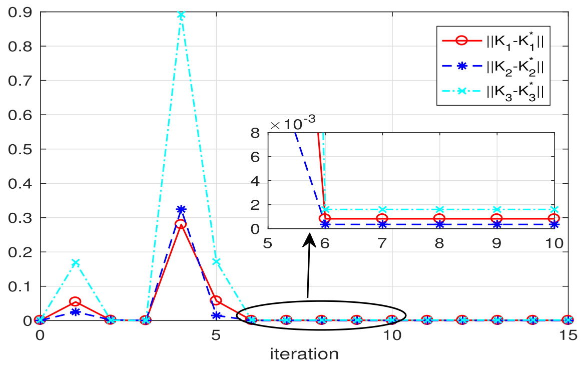

FIGURE 3. Case 1: Convergence of K_i in on-policy game Q-learning.

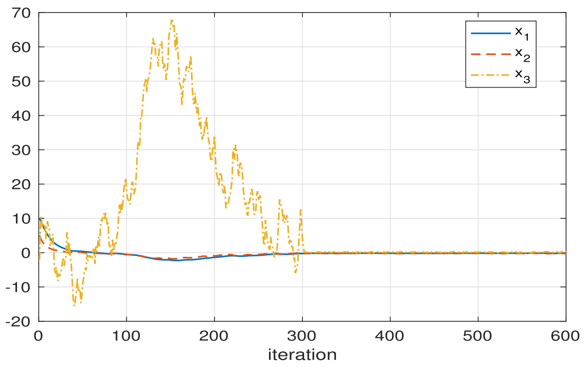

FIGURE 4. Case 1: The system states x in on-policy game Q-learning.

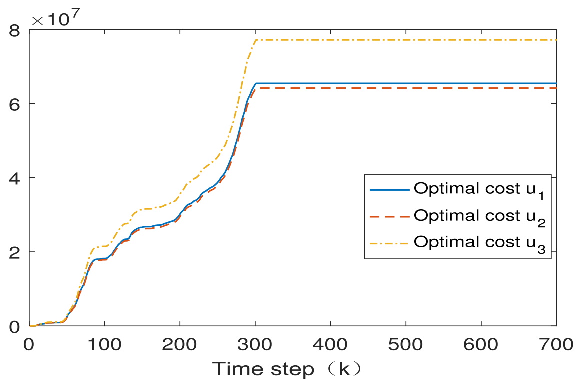

FIGURE 5. Case 1: The optimal costs in on-policy game Q-learning.

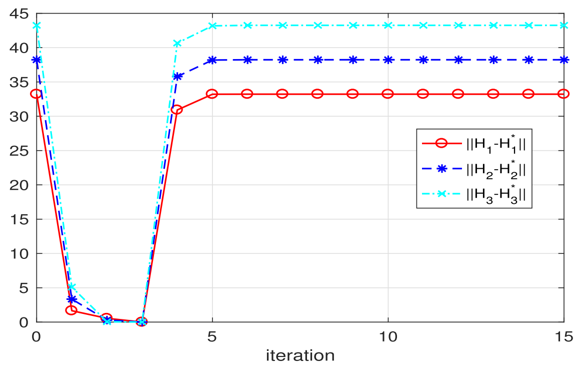

FIGURE 6. Case 2: Convergence of H_i in on-policy game Q-learning.

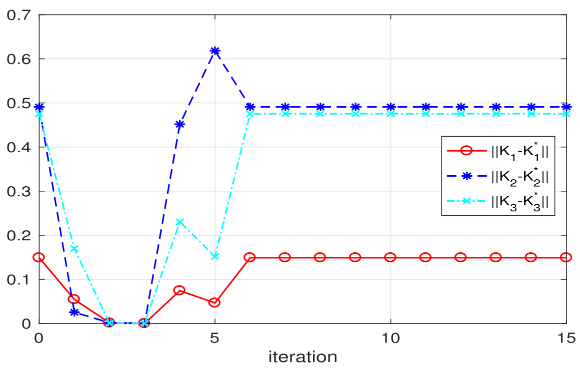

FIGURE 7. Case 2: Convergence of K_i in on-policy game Q-learning.

K_i^* under the probing noises Case 2 during the implementing of on-policy game Q-learning Algorithm 1. Fig. 8 shows the state responses of the game system.

Under the probing noises Case 3, Fig. 9 and Fig. 10 show the errors of matrices H_i and H_i^*, and the errors of controller gains K_i and K_i^*, Fig. 11 shows the state responses of the game system.

Simulation results when using the off-policy game Q-learning Algorithm 2: Since there is no bias of solution (H_i, K_i) caused by adding probing noises, then the convergence results of (H_i, K_i) only under the probing noises Case 1 are plotted Fig. 12 and Fig. 13. Fig. 14 shows the state responses of the game system. The performance J_i along the system trajectories under the learned optimal control policies are plotted in Fig. 15.

B. SIMULATION RESULTS FOR THE FIVE-PLAYER SYSTEM

Consider the following linear DT system with five players, which play non-zero sum game.

$$x_{k+1} = Ax_k + B_1u_1 + B_2u_2 + B_3u_3 + B_4u_4 + B_5u_5 \quad (45)$$

where

$$A = \begin{bmatrix} 0.906488 & 0.0816012 & -0.0005 \\ 0.074349 & 0.90121 & -0.000708383 \\ 0 & 0 & 0.132655 \end{bmatrix}$$

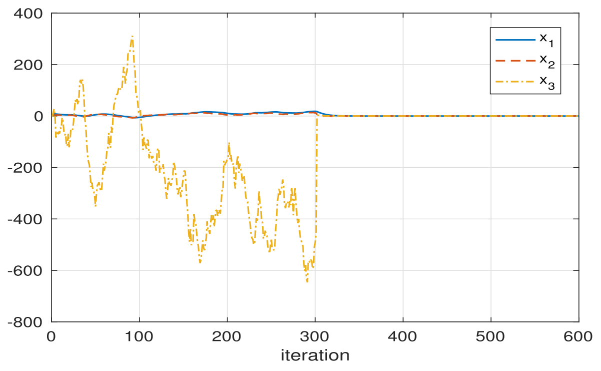

FIGURE 8. Case 2: The system states x in on-policy game Q-learning.

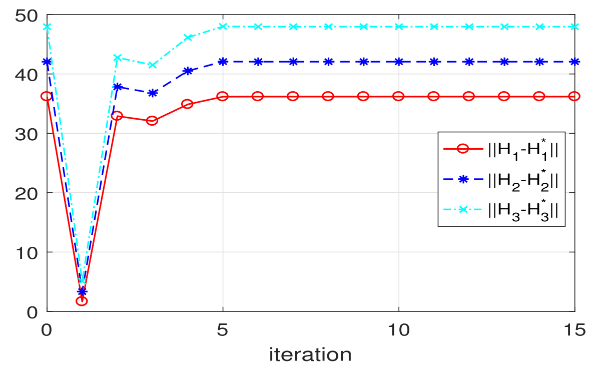

FIGURE 9. Case 3: Convergence of H_i in on-policy game Q-learning.

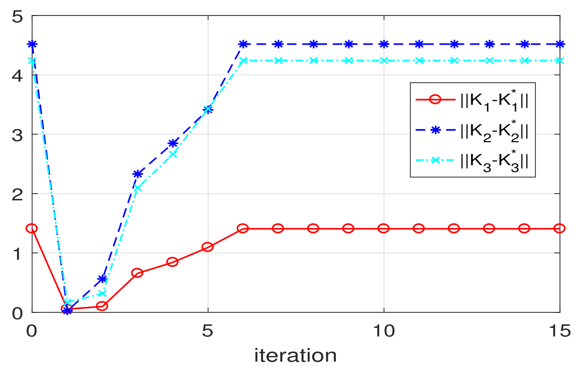

FIGURE 10. Case 3: Convergence of K_i in on-policy game Q-learning.

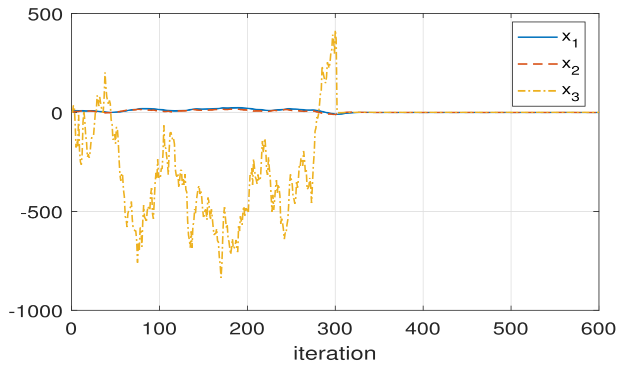

FIGURE 11. Case 3: The system states x in on-policy game Q-learning.

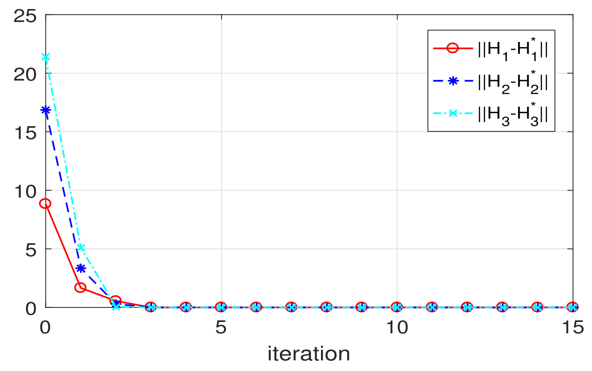

FIGURE 12. Convergence of H_i in off-policy game Q-learning.

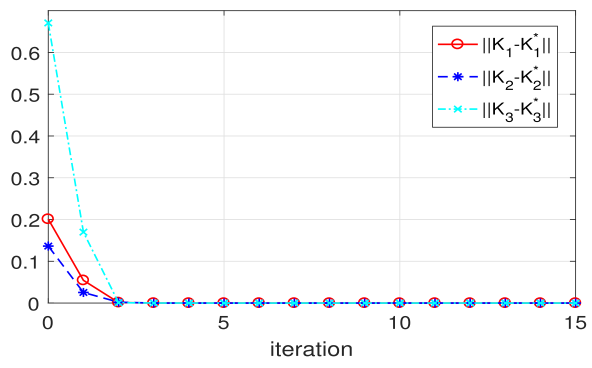

FIGURE 13. Convergence of K_i in off-policy game Q-learning.

$$B_1 = \begin{bmatrix} -0.00150808 \\ -0.0096 \\ 0.867345 \end{bmatrix}, \quad B_2 = \begin{bmatrix} 0.00951892 \\ 0.00038373 \\ 0 \end{bmatrix}$$

$$B_3 = \begin{bmatrix} -0.00563451 \\ -0.08962 \\ 0.356478 \end{bmatrix}, \quad B_4 = \begin{bmatrix} 0.0123956 \\ 0.068 \\ -0.05673 \end{bmatrix}$$

$$B_5 = \begin{bmatrix} -0.125 \\ 0.4 \\ -0.4898 \end{bmatrix}$$

Choose $Q_1 = diag(4,4,4)$, $Q_2 = diag(5,5,5)$, $Q_3 = diag(6,6,6)$, $Q_4 = diag(7,7,7)$, $Q_5 = diag(3,3,3)$ and $R_1 = R_2 = R_3 = R_4 = R_5 = 1$. Rewrite (18) as

$$H_i^* = \Lambda_i + \begin{bmatrix} A & B_1 & \dots & B_5 \\ K_1A & K_1B_1 & \dots & K_1B_5 \\ K_2A & K_2B_1 & \dots & K_2B_5 \\ K_3A & K_3B_1 & \dots & K_3B_5 \\ K_4A & K_4B_1 & \dots & K_4B_5 \\ K_5A & K_5B_1 & \dots & K_5B_5 \end{bmatrix}^T \times H_i^* \begin{bmatrix} A & B_1 & \dots & B_5 \\ K_1A & K_1B_1 & \dots & K_1B_5 \\ K_2A & K_2B_1 & \dots & K_2B_5 \\ K_3A & K_3B_1 & \dots & K_3B_5 \\ K_4A & K_4B_1 & \dots & K_4B_5 \\ K_5A & K_5B_1 & \dots & K_5B_5 \end{bmatrix} \quad (46)$$

The model-based iterative algorithm in terms of (46) is used in MATLAB to obtain the real solution. Thus, the optimal Q-function matrices (H_1^*, H_2^*, H_3^*, H_4^*, H_5^*) and the optimal controller gains (K_1^*, K_2^*, K_3^*, K_4^*, K_5^*) can be obtained.

$$H_1^* = \begin{bmatrix} 25.4360 & 9.7277 & -0.0168 & -0.1127 & 0.2217 & -0.9082 & 0.8767 & 0.6170 \\ 9.7277 & 14.1358 & -0.0049 & -0.0663 & 0.0981 & -0.9637 & 0.8233 & 2.8781 \\ -0.0168 & -0.0049 & 4.0707 & 0.4623 & -0.0002 & 0.1904 & -0.0307 & -0.2603 \\ -0.1127 & -0.0663 & 0.4623 & 4.0228 & -0.0012 & 1.2483 & -0.2034 & -1.7168 \\ 0.2217 & 0.0981 & -0.0002 & -0.0012 & 1.0023 & -0.0091 & 0.0089 & 0.0052 \\ -0.9082 & -0.9637 & 0.1904 & 1.2483 & -0.0091 & 1.6022 & -0.1594 & -0.9780 \\ 0.8767 & 0.8233 & -0.0307 & -0.2034 & 0.0089 & -0.1594 & 1.0804 & 0.3301 \\ 0.6170 & 2.8781 & -0.2603 & -1.7168 & 0.0052 & -0.9780 & 0.3301 & 3.1613 \end{bmatrix}$$

$$H_2^* = \begin{bmatrix} 29.7586 & 10.9982 & -0.0182 & -0.1208 & 0.2561 & -1.0225 & 0.9944 & 0.6000 \\ 10.9982 & 16.7580 & -0.0058 & -0.0778 & 0.1107 & -1.1192 & 0.9532 & 3.3906 \\ -0.0182 & -0.0058 & 5.0883 & 0.5775 & -0.0002 & 0.2378 & -0.0383 & -0.3254 \\ -0.1208 & -0.0778 & 0.5775 & 4.7758 & -0.0012 & 1.5589 & -0.2536 & -2.1457 \\ 0.2561 & 0.1107 & -0.0002 & -0.0012 & 1.0027 & -0.0103 & 0.0101 & 0.0048 \\ -1.0225 & -1.1192 & 0.2378 & 1.5589 & -0.0103 & 1.7443 & -0.1920 & -1.2033 \\ 0.9944 & 0.9532 & -0.0383 & -0.2536 & 0.0101 & -0.1920 & 1.0941 & 0.3960 \\ 0.6000 & 3.3906 & -0.3254 & -2.1457 & 0.0048 & -1.2033 & 0.3960 & 3.6386 \end{bmatrix}$$

$$H_3^* = \begin{bmatrix} 34.0812 & 12.2688 & -0.0196 & -0.1290 & 0.2906 & -1.1368 & 1.1121 & 0.5830 \\ 12.2688 & 19.3802 & -0.0066 & -0.0893 & 0.1234 & -1.2747 & 1.0830 & 3.9032 \\ -0.0196 & -0.0066 & 6.1059 & 0.6926 & -0.0002 & 0.2852 & -0.0459 & -0.3906 \\ -0.1290 & -0.0893 & 0.6926 & 5.5288 & -0.0013 & 1.8694 & -0.3037 & -2.5747 \\ 0.2906 & 0.1234 & -0.0002 & -0.0013 & 1.0030 & -0.0114 & 0.0112 & 0.0044 \\ -1.1368 & -1.2747 & 0.2852 & 1.8694 & -0.0114 & 1.8864 & -0.2247 & -1.4285 \\ 1.1121 & 1.0830 & -0.0459 & -0.3037 & 0.0112 & -0.2247 & 1.1079 & 0.4620 \\ 0.5830 & 3.9032 & -0.3906 & -2.5747 & 0.0044 & -1.4285 & 0.4620 & 4.1159 \end{bmatrix}$$

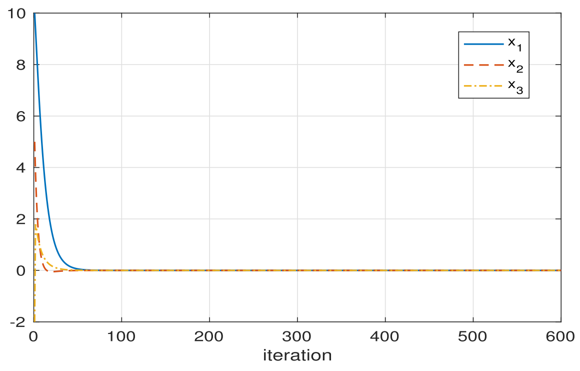

FIGURE 14. The system states *x* in off-policy game Q-learning.

$$H_4^* = \begin{bmatrix} 38.4038 & 13.5393 & -0.0210 & -0.1372 & 0.3250 & -1.2512 & 1.2297 & 0.5660 \\ 13.5393 & 22.0024 & -0.0074 & -0.1009 & 0.1361 & -1.4302 & 1.2129 & 4.4158 \\ -0.0210 & -0.0074 & 7.1235 & 0.8078 & -0.0002 & 0.3326 & -0.0535 & -0.4557 \\ -0.1372 & -0.1009 & 0.8078 & 6.2818 & -0.0014 & 2.1800 & -0.3539 & -3.0037 \\ 0.3250 & 0.1361 & -0.0002 & -0.0014 & 1.0034 & -0.0126 & 0.0124 & 0.0040 \\ -1.2512 & -1.4302 & 0.3326 & 2.1800 & -0.0126 & 2.0285 & -0.2573 & -1.6537 \\ 1.2297 & 1.2129 & -0.0535 & -0.3539 & 0.0124 & -0.2573 & 1.1216 & 0.5280 \\ 0.5660 & 4.4158 & -0.4557 & -3.0037 & 0.0040 & -1.6537 & 0.5280 & 4.5932 \end{bmatrix}$$

$$H_5^* = \begin{bmatrix} 21.1134 & 8.4572 & -0.0154 & -0.1045 & 0.1872 & -0.7938 & 0.7590 & 0.6340 \\ 8.4572 & 11.5136 & -0.0041 & -0.0548 & 0.0854 & -0.8082 & 0.6935 & 2.3655 \\ -0.0154 & -0.0041 & 3.0531 & 0.3471 & -0.0002 & 0.1430 & -0.0231 & -0.1951 \\ -0.1045 & -0.0548 & 0.3471 & 3.2698 & -0.0011 & 0.9378 & -0.1532 & -1.2878 \\ 0.1872 & 0.0854 & -0.0002 & -0.0011 & 1.0019 & -0.0080 & 0.0077 & 0.0056 \\ -0.7938 & -0.8082 & 0.1430 & 0.9378 & -0.0080 & 1.4600 & -0.1268 & -0.7528 \\ 0.7590 & 0.6935 & -0.0231 & -0.1532 & 0.0077 & -0.1268 & 1.0667 & 0.2641 \\ 0.6340 & 2.3655 & -0.1951 & -1.2878 & 0.0056 & -0.7528 & 0.2641 & 2.6840 \end{bmatrix}$$

$$\begin{aligned} K_1^* &= \begin{bmatrix} -0.1559 & -0.4921 & -0.1018 \end{bmatrix} \\ K_2^* &= \begin{bmatrix} -0.2372 & -0.0944 & -0.0006 \end{bmatrix} \\ K_3^* &= \begin{bmatrix} 0.8838 & 0.5288 & -0.0597 \end{bmatrix} \\ K_4^* &= \begin{bmatrix} -1.1014 & -0.7778 & 0.0084 \end{bmatrix} \\ K_5^* &= \begin{bmatrix} 0.3424 & -0.7149 & -0.0138 \end{bmatrix} \end{aligned} \tag{47}$$

Simulation results when using off-policy game Q-learning Algorithm 2: Under the probing noises Case 1, Fig. 16 and

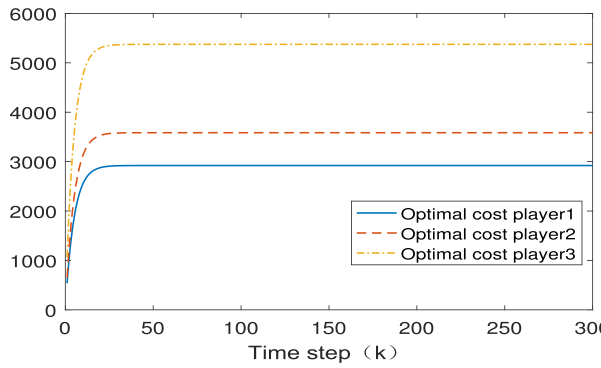

FIGURE 15. The optimal costs in off-policy game Q-learning.

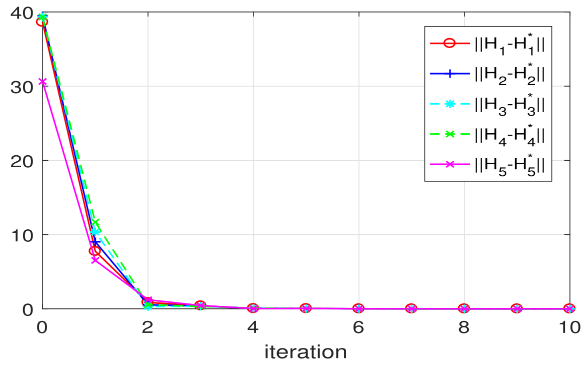

FIGURE 16. Convergence of H_i in off-policy game Q-learning.

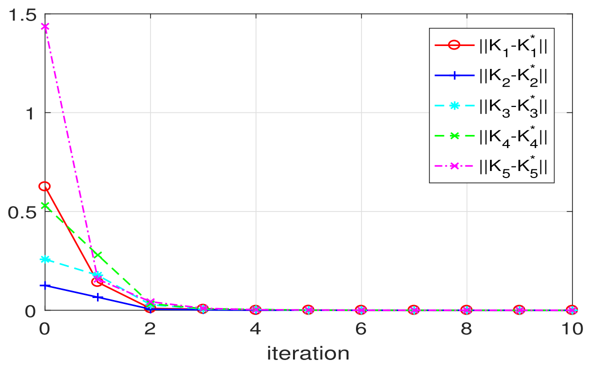

FIGURE 17. Convergence of K_i in off-policy game Q-learning.

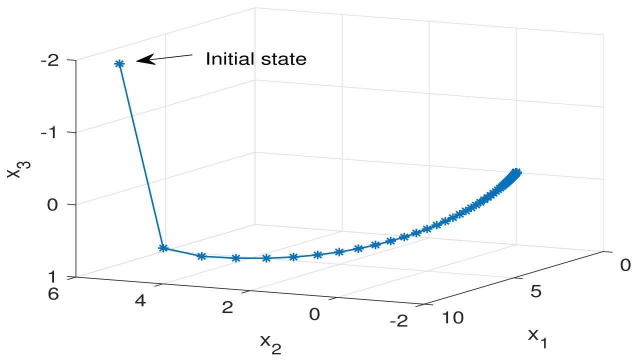

FIGURE 18. The system states x in off-policy game Q-learning.

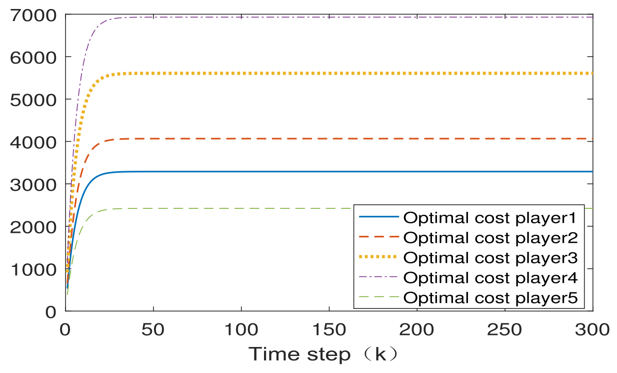

FIGURE 19. The optimal costs in off-policy game Q-learning.

Fig. 17 show the convergences of the Q-function matrices H_i and the learned control gains K_i. Fig. 18 shows the state responses of the game system. The above three figures are obtained by implementing the proposed off-policy game Q-learning algorithm. The performance J_i is plotted in Fig. 19 under the learned Nash equilibrium solution.

C. RESULTS ANALYSIS AND COMPARISONS

From Table 1, one can find that solution deviation indeed existed when using the on-policy game Q-learning for multi-player games, while there is no bias of solution of Q-function matrices H_i and the optimal controller gains K_i when implementing the off-policy game Q-learning for the multi-player games. With the increasing of probing noises, the state response in Fig. 4, 8 and 11 and the cost in Fig. 5 is more remarkably affected since the system is disturbed by adding probing noises. However, as shown in Fig. 14 and 15 for the case of three-player games and Fig. 18 and 19 for the case of five-player games, the systems converge with fast velocity and without overshoot, and the costs are smaller than those using the on-policy game Q-learning algorithm.

VII. CONCLUSION

In this paper, an off-policy game Q-learning algorithm is proposed to solve multi-player non-zero sum game problems in linear DT systems without knowing the dynamics of models. The probing noises are added into control inputs during learning solutions and the convergence and unbiasedness of the proposed off-policy game Q-learning are presented with rigorously theoretical proofs. Simulation results have demonstrated the effectiveness of the proposed method.

REFERENCES

[1] F. L. Lewis and D. Liu, "Reinforcement learning and approximate dynamic programming for feedback control," *IEEE Circuits Syst. Mag.*, vol. 9, no. 3, pp. 32–50, Aug. 2009.
[2] M. Riedmiller, "Neural fitted *Q* iteration—First experiences with a data efficient neural reinforcement learning method," in *Proc. Eur. Conf. Mach. Learn.*, 2005, pp. 317–328.
[3] J. Fu, H. He, and X. Zhou, "Adaptive learning and control for MIMO system based on adaptive dynamic programming," *IEEE Trans. Neural Netw.*, vol. 22, no. 7, pp. 1133–1148, Jul. 2011.
[4] R. Munos, T. Stepleton, A. Harutyunyan, and M. Bellemare, "Safe and efficient off-policy reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 1054–1062.
[5] J. Li, H. Modares, T. Chai, F. L. Lewis, and L. Xie, "Off-policy reinforcement learning for synchronization in multiagent graphical games," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2434–2445, Oct. 2017.

[6] J. Li, B. Kiumarsi, T. Chai, F. L. Lewis, and J. Fan, "Off-policy reinforcement learning: Optimal operational control for two-time-scale industrial processes," *IEEE Trans. Cybern.*, vol. 47, no. 12, pp. 4547–4558, Dec. 2017.

[7] J. Li, T. Chai, F. L. Lewis, J. Fan, Z. Ding, and J. Ding, "Off-policy Q-learning: Set-point design for optimizing dual-rate rougher flotation operational processes," *IEEE Trans. Ind. Electron.*, vol. 65, no. 5, pp. 4092–4102, May 2018.

[8] B. Kiumarsi, F. L. Lewis, and Z.-P. Jiang, "H_∞ control of linear discrete-time systems: Off-policy reinforcement learning," *Automatica*, vol. 78, pp. 144–152, Apr. 2017.

[9] R. Song, F. L. Lewis, Q. Wei, and H. Zhang, "Off-policy actor-critic structure for optimal control of unknown systems with disturbances," *IEEE Trans. Cybern.*, vol. 46, no. 5, pp. 1041–1050, May 2015.

[10] B. Luo, H.-N. Wu, and T. Huang, "Off-policy reinforcement learning for H_∞ control design," *IEEE Trans. Cybern.*, vol. 45, no. 1, pp. 65–76, Jan. 2015.

[11] B. Luo, H.-N. Wu, T. Huang, and D. Liu, "Data-based approximate policy iteration for affine nonlinear continuous-time optimal control design," *Automatica*, vol. 50, no. 12, pp. 3281–3290, 2014.

[12] H. Modares, F. L. 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, Oct. 2015.

[13] H. Modares and F. L. Lewis, "Optimal tracking control of nonlinear partially-unknown constrained-input systems using integral reinforcement learning," *Automatica*, vol. 50, no. 7, pp. 1780–1792, Jul. 2014.

[14] H. Zhang, Q. Wei, and Y. Luo, "A novel infinite-time optimal tracking control scheme for a class of discrete-time nonlinear systems via the greedy HDP iteration algorithm," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 38, no. 4, pp. 937–942, Aug. 2008.

[15] Q. Wei and D. Liu, "Adaptive dynamic programming for optimal tracking control of unknown nonlinear systems with application to coal gasification," *IEEE Trans. Autom. Sci. Eng.*, vol. 11, no. 4, pp. 1020–1036, Oct. 2014.

[16] H. Zhang, R. Song, Q. Wei, and T. Zhang, "Optimal tracking control for a class of nonlinear discrete-time systems with time delays based on heuristic dynamic programming," *IEEE Trans. Neural Netw.*, vol. 22, no. 12, pp. 1851–1862, Dec. 2011.

[17] B. Kiumarsi, F. L. Lewis, M.-B. Naghibi-Sistani, and A. Karimpour, "Optimal tracking control of unknown discrete-time linear systems using input-output measured data," *IEEE Trans. Cybern.*, vol. 45, no. 12, pp. 2770–2779, Dec. 2015.

[18] Q. Wei, F. L. Lewis, Q. Sun, P. Yan, and R. Song, "Discrete-time deterministic Q-learning: A novel convergence analysis," *IEEE Trans. Cybern.*, vol. 47, no. 5, pp. 1224–1237, May 2017.

[19] J. Y. Lee, J. B. Park, and Y. H. Choi, "Integral Q-learning and explorized policy iteration for adaptive optimal control of continuous-time linear systems," *Automatica*, vol. 48, no. 11, pp. 2850–2859, Nov. 2012.

[20] 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.

[21] 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_∞ control," *Automatica*, vol. 43, no. 3, pp. 473–481, Mar. 2007.

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

[23] J. H. Kim and F. L. Lewis, "Model-free H_∞ control design for unknown linear discrete-time systems via Q-learning with LMI," *Automatica*, vol. 46, no. 8, pp. 1320–1326, Aug. 2010.

[24] Y. Jiang, J. Fan, T. Chai, F. L. Lewis, and J. N. Li, "Tracking control for linear discrete-time networked control systems with unknown dynamics and dropout," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 10, pp. 4607–4620, Oct. 2018.

[25] K. G. Vamvoudakis, "Q-learning for continuous-time linear systems: A model-free infinite horizon optimal control approach," *Syst. Control Lett.*, vol. 100, pp. 14–20, Feb. 2017.

[26] K. G. Vamvoudakis, "Q-learning for continuous-time graphical games on large networks with completely unknown linear system dynamics," *Int. J. Robust Nonlinear Control*, vol. 27, no. 16, pp. 2900–2920, Nov. 2017.

[27] B. Luo, D. Liu, T. Huang, and D. Wang, "Model-free optimal tracking control via critic-only Q-learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 10, pp. 2134–2144, Oct. 2016.

[28] B. Luo, D. Liu, H.-N. Wu, D. Wang, and F. L. Lewis, "Policy gradient adaptive dynamic programming for data-based optimal control," *IEEE Trans. Cybern.*, vol. 47, no. 10, pp. 3341–3354, Oct. 2017.

[29] J. Li, T. Chai, F. L. Lewis, Z. Ding, and Y. Jiang, "Off-policy interleaved Q-learning: Optimal control for affine nonlinear discrete-time systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 5, pp. 1308–1320, May 2019.

[30] K. G. Vamvoudakis, H. Modares, B. Kiumarsi, and F. L. Lewis, "Game theory-based control system algorithms with real-time reinforcement learning: How to solve multiplayer games online," *IEEE Control Syst.*, vol. 37, no. 1, pp. 33–52, Feb. 2017.

[31] B. Kiumarsi, F. L. Lewis, H. Modares, A. Karimpour, and M.-B. Naghibi-Sistani, "Reinforcement Q-learning for optimal tracking control of linear discrete-time systems with unknown dynamics," *Automatica*, vol. 50, no. 4, pp. 1167–1175, 2014.

[32] 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, Aug. 2011.

[33] K. G. Vamvoudakis, F. L. Lewis, and G. R. Hudas, "Multi-agent differential graphical games: Online adaptive learning solution for synchronization with optimality," *Automatica*, vol. 48, no. 8, pp. 1598–1611, 2012.

JINNA LI (M'12) received the M.S. and Ph.D. degrees from Northeastern University, Shenyang, China, in 2006 and 2009, respectively. From April 2009 to April 2011, she held a postdoctoral position at the Lab of Industrial Control Networks and Systems, Shenyang Institute of Automation, Chinese Academy of Sciences. From June 2014 to June 2015, she was a Visiting Scholar granted by the China Scholarship Council with Energy Research Institute, Nanyang Technological University, Singapore. From September 2015 to June 2016, she was a Domestic Young Core Visiting Scholar granted by the Ministry of Education of China with the State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University. From January 2017 to July 2017, she was a Visiting Scholar with the School of Electrical and Electronic Engineering, The University of Manchester, U.K. She is currently a Full Professor with the School of Information and Control Engineering, Liaoning Shihua University. She has also been with the State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University. Her current research interests include neural networks, reinforcement learning, optimal operational control, distributed optimization control, and data-based control.

ZHENFEI XIAO received the B.S. degree in electrical engineering and automation from Great Wall College, China University of Geosciences, Baoding, China, in 2018. He is currently pursuing the M.S. degree in control theory and control engineering with Liaoning Shihua University, Fushun, China. His current research interests include neural networks, reinforcement learning, and data-based control.

PING LI (SM'08) received the Ph.D. degree in automatic control from Zhejiang University, Hangzhou, China, in 1995. He is currently a Full Professor with the School of Information and Control Engineering, Liaoning Shihua University. He has published over 100 articles and completed over 20 research projects supported by national and ministry funds. He has received more than ten awards of nation and ministry. His research interests include process control and automation, especially the advanced control and optimization of chemical industry process control systems.