

# Data-Driven Wind Farm Control via Multiplayer Deep Reinforcement Learning

Hongyang Dong<sup>1b</sup>, *Member, IEEE*, and Xiaowei Zhao<sup>1b</sup>, *Member, IEEE*

**Abstract**—This brief proposes a novel data-driven control scheme to maximize the total power output of wind farms subject to strong aerodynamic interactions among wind turbines. The proposed method is model-free and has strong robustness, adaptability, and applicability. Particularly, distinct from the state-of-the-art data-driven wind farm control methods that commonly use the steady-state or time-averaged data (such as turbines' power outputs under steady wind conditions or from steady-state models) to carry out learning, the proposed method directly mines in-depth the time-series data measured at turbine rotors under time-varying wind conditions to achieve farm-level power maximization. The control scheme is built on a novel multiplayer deep reinforcement learning method (MPDRL), in which a special critic-actor-distractor structure, along with deep neural networks (DNNs), is designed to handle the stochastic feature of wind speeds and learn optimal control policies subject to a user-defined performance metric. The effectiveness, robustness, and scalability of the proposed MPDRL-based wind farm control method are tested by prototypical case studies with a dynamic wind farm simulator (WFSim). Compared with the commonly used greedy strategy, the proposed method leads to clear increases in farm-level power generation in case studies.

**Index Terms**—Machine learning, reinforcement learning, wind energy, wind farm control, wind turbine control.

## I. INTRODUCTION

AS ONE of the most efficient forms of green energy, wind power plays a key role in the global effort toward net-zero emissions. Particularly, 15 large offshore wind farms were put into operation in 2020, with an average capacity of 347 MW [1]. Though single turbine's control strategies have been widely studied, directly using these methods to control every turbine in a wind farm via a noncooperative way can lead to significantly degraded operating efficiency. A lot of studies [2], [3], [4] have demonstrated that the greedy strategy (i.e., every turbine in the farm aims to maximize its own power outputs) is not the optimal control strategy for farm-level power generation maximization. This phenomenon is caused by the aerodynamic couplings among turbines—the wakes induced by upstream turbines can severely influence the power generation of downstream turbines, which is commonly mentioned as the wake effect in the literature. Therefore, the

farm-level control strategies should be considered to operate all the turbines cooperatively to mitigate wake effects and increase the whole farm's power generation.

Compared with single-turbine cases, the main challenges of farm-level control tasks come from the complex aerodynamics and stochastic properties of wakes. On one hand, wakes are hard to be accurately modeled, bringing barriers to controller design. On the other hand, control-induced wake change has time-delayed features—it typically requires tens/hundreds of seconds to propagate from upstream turbines to downstream turbines, resulting in difficulties in evaluating control performance, especially from a relatively long-term standpoint. Many studies used simplified wake models [2], [5], [6] to evaluate steady-state wake locations and estimate the effective flow velocities and power outputs at each turbine. Then these estimations were used to decide optimal control actions. For example, a famous steady-state parametric model, referred to as FLORIS, was proposed in [2] to achieve yaw setting optimization. Though such designs are easy to implement, their performance can be severely influenced by the model accuracy, and they have difficulties in achieving closed-loop wind farm control under uncertain environments [3]. Some studies used dynamic models that typically have higher fidelity than steady-state models to develop wind farm control methods and achieve better control performance. For example, a model predictive control (MPC) method was proposed in [7] based on a control-oriented dynamic wind farm model developed in [8]. This meaningful method was shown to be effective in increasing the whole farm's power generation. A small limitation is that it used a relatively large number of states (tens of thousands or even hundreds of thousands) in receding-horizon control. As an extension study, Chen et al. [9] reduced the number of system states via deep autoencoder and carried out MPC based on the reduced-order model. However, those elegant MPC methods still rely on the accuracy of the underlying wind flow and wind farm models, and their performance could be degraded due to modeling errors and unmodeled dynamics.

In summary, the model-based wind farm control methods have shown good effectiveness and are usually easy to implement, but they may suffer from high system complexity, inevitable modeling inaccuracy, and stochastic environment uncertainty. In recent years, data-driven and model-free wind farm methods have drawn extensive research interest and have been treated as promising candidates to overcome the limitations of their model-based counterparts. For the wind farm power maximization tasks, a game-theoretic method was proposed in [10] to decide the optimal induction factors for every turbine in the farm. Park and Law [11] proposed a

Manuscript received 31 August 2022; accepted 11 November 2022. This work was supported by the U.K. Engineering and Physical Sciences Research Council under Grant EP/R007470/1 and Grant EP/S000747/1. Recommended by Associate Editor P. F. Odgaard. (*Corresponding author: Xiaowei Zhao.*)

The authors are with the Intelligent Control and Smart Energy (ICSE) Research Group, School of Engineering, University of Warwick, CV4 7AL Coventry, U.K. (e-mail: hongyang.dong@warwick.ac.uk; xiaowei.zhao@warwick.ac.uk).

Color versions of one or more figures in this article are available at <https://doi.org/10.1109/TCST.2022.3223185>.

Digital Object Identifier 10.1109/TCST.2022.3223185

1063-6536 © 2022 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.

See <https://www.ieee.org/publications/rights/index.html> for more information.

Bayesian-based searching method for yaw settings' optimization. A deep reinforcement learning (DRL)-based wind farm control method was proposed in [12], in which an internal wake model was used to guide the learning process. However, all these important results are based on steady wind conditions, and they can only provide fixed yaw and/or induction factor settings subject to unchanged wind speeds. A DRL-based yaw control method that aimed to handle stepwise-varying inflow conditions was designed in [13], but it was built upon a steady-state wind farm model. Another two wind farm control approaches via DRL were proposed in [14] and [15]. They are model-free and can adapt to time-varying wind speeds. But a limitation is that they used averaged power outputs over particular time spans to construct rewards. In addition, they only considered yaw control tasks. To sum up, though data-driven wind farm control methods have aroused extensive attention, there are still many essential issues that limit the applicability and feasibility of current results, including the reliance on steady-state or time-averaged data (such as turbines' power outputs under steady wind conditions or from steady-state models), lack of robustness to time-varying wind conditions, and insufficiency in data mining.

Aiming to address the aforementioned challenges in current approaches, this brief proposes a new intelligent wind farm control method via a novel DRL algorithm. DRL [16], [17], [18] is a booming artificial intelligence technique that is currently triggering the technological revolution of many industrial sectors. It is capable of fulfilling high-performance data-driven control for multiple tasks of a large class of complex systems. In this brief, an application-oriented DRL method is designed to maximize wind farm power generation by yaw and induction control. Particularly, a multiplayer DRL (MPDRL) algorithm is designed to learn the optimal control policy subject to a user-defined performance metric using the available measurements at turbine rotors, and deep neural networks (DNNs) are used in it as information processors and universal approximators. Its effectiveness is verified by a dynamic wind farm simulator (WFSim) [8] for wind farms with different specifications. The test results show that our method can achieve clear farm-level power generation increases (over 30% in case studies) compared with the benchmark. Further explanations regarding the novelty of the proposed method in comparison to relevant studies are discussed as follows.

- 1) Unlike mainstream wind farm control approaches [2], [3], [5], [6], [7], [19], our method relaxes the dependence on wind farm models. Using the merits of DNN and DRL, it has the ability to capture the core system information and maximize the farm-level power production only by data. Our method has strong robustness against unmodeled dynamics and stochastic environments.
- 2) Compared with important data-driven wind farm control methods (including some results based on DRL) [10], [11], [12], [13], [20], no steady-state or time-averaged data are required to carry out the learning process in our design. Instead, by embedding DNNs that can handle time-series data (such as long-short-term-memory (LSTM) networks, gated recurrent units (GRU), and

transformer networks) into the main DRL structure, our MPDRL can mine in-depth the data measured at turbine rotors, handle wake propagation delay, and achieve closed-loop wind farm control under time-varying wind conditions. These essential features render our method to have enhanced performance.

- 3) Distinct from the most recent DRL-based wind farm control approaches [12], [13], [14], [15], a special distractor network is used in our method. This design not only mitigates the non-Markovian feature induced by the stochastic nature of wind conditions but also enhances our method's robustness.

It is noteworthy that the method proposed in this brief is partially built upon [4], [14], [15], and substantial new contributions have been made. Specifically, though Dong et al. [14] and Dong and Zhao [15] also designed DRL algorithms to optimize farm-level power generation, they used averaged power outputs over particular time spans to construct rewards, leading to potentially limited adaptability and robustness. In addition, they only considered yaw control tasks. These limitations are fully addressed in this work. The method proposed in this brief does not rely on any averaged data. Instead, time-series data that are easy to measure are used to help our *RL* agent mine system information and adapt to real-time changes in environmental conditions. Moreover, not only yaw control but also axial-induction-based control strategies are considered in this brief. There are also several essential differences between the work in this brief and Dong and Zhao [4]. First, these two studies consider different tasks. Instead of achieving power generation maximization, Dong and Zhao [4] focused on providing ancillary services by adjusting induction factors. The yaw control strategy was not designed in it, which is vital for the farm-level power generation maximization tasks. In contrast, the present brief develops a DRL structure for both yaw and induction control to maximize the whole farm's power generation. These two kinds of tasks have quite different features and inherent control system design logic. It should be emphasized that simultaneously achieving yaw and induction control is challenging due to the distinctive and incompatible features of yaw angles and induction-related states. We use the multiplayer (MP) concept to address this issue. Our MP-DRL algorithm has two actors to learn the optimal yaw control and induction control policies simultaneously yet separately, one distractor to evaluate the worst case external disturbances (i.e., wind speed changes) to the performance metric, and one critic to learn the optimal long-term performance functional. Such a special DRL structure allows us to not only handle the incompatibility of yaw and induction control but also bring robustness against external disturbances. Based on these facts, the proposed MPDRL-based wind farm control method is more general and flexible, and it renders enhanced adaptability and robustness compared with the results in [14], [15] and [4].

In the remainder of this brief, we introduce the farm-level power maximization task in Section II. Following that, the MPDRL wind farm control method is explained in Section III. To verify its effectiveness, simulations with a dynamic WFSim for different wind farms are presented in Section IV. After that, we conclude the whole work in Section V.

## II. PROBLEM FORMALIZATION

We consider a wind farm with  $n$  turbines that are denoted by  $\mathcal{WT}_1, \mathcal{WT}_2, \dots, \mathcal{WT}_n$ , respectively. It is well-understood that the power captured by a turbine  $\mathcal{WT}_i$  is directly related to its inflow wind speed  $U_i$ , the induction-related state  $\alpha_i$  (such as the induction factor or some other states that are related to the induction factor, e.g., the modified thrust coefficient), and yaw offset  $\beta_i$  [2], [10], [11], formalized by

$$E_i = h(U_i, \alpha_i, \beta_i). \quad (1)$$

For turbine-level control, the induction-related state  $\alpha_i$  decides the blade pitch angle and rotor torque. It should be emphasized that in this brief, the specific expression of  $h$  is not required in controller design. Based on (1), the whole farm's power is

$$E = \sum_{i=1}^n E_i = \sum_{i=1}^n h(U_i, \alpha_i, \beta_i). \quad (2)$$

Before introducing the specific wind farm control objectives considered in this brief, we first define control inputs, states, and exogenous inputs (i.e., disturbances) of the system. The control inputs are the changes in the induction-related states and yaw angles of all the turbines, denoted by  $\delta\alpha = [\delta\alpha_1, \delta\alpha_2, \dots, \delta\alpha_n]$  and  $\delta\beta = [\delta\beta_1, \delta\beta_2, \dots, \delta\beta_n]$ , respectively. Then we define a regularized state vector to be

$$\bar{x} = [\bar{\alpha}_1, \bar{\beta}_1, \bar{E}_1, \bar{\alpha}_2, \bar{\beta}_2, \bar{E}_2, \dots, \bar{\alpha}_n, \bar{\beta}_n, \bar{E}_n] \quad (3)$$

with  $\bar{\alpha}_i = ((\alpha_i - \alpha_{\max}) + (\alpha_i - \alpha_{\min})) / (\alpha_{\max} - \alpha_{\min})$ ,  $\bar{\beta}_i = ((\beta_i - \beta_{\max}) + (\beta_i - \beta_{\min})) / (\beta_{\max} - \beta_{\min})$ , and  $\bar{E}_i = (E_i / E_r)$ . Here,  $\alpha_{\max}$  and  $\alpha_{\min}$  are the upper and lower bounds of induction-related states, respectively;  $\beta_{\max}$  and  $\beta_{\min}$  are the upper and lower bounds of yaw angles, respectively; and  $E_r$  is the turbine's rated power. Therefore, one has  $\bar{\beta}_i \in [-1, 1]$  and  $\bar{\alpha}_i \in [-1, 1]$ . We also define the regularized exogenous input:

$$w = [\bar{U}_1, \bar{U}_2, \dots, \bar{U}_n]^T \quad (4)$$

where  $\bar{U}_i = 2U_i / U_N - 1$  with  $U_N$  is a user-defined constant for normalization purposes.

The wind farm controller considered in this brief should make a trade-off for the following two objectives subject to wake effects: 1) maximizing the farm-level power generation—this is the main objective and 2) avoiding large loads caused by induction and yaw control.

Based on these objectives, we define the following reward function for time step  $k$ :

$$r(k) = -c_1 \sum_{i=1}^n \bar{E}_i(k) + \frac{c_2}{n} \sum_{i=1}^n |F_i(k) - F_i(k-1)| + \frac{c_3}{n} \sum_{i=1}^n \left| \frac{\beta_i(k)}{b_\beta} \right| \quad (5)$$

and here  $c_1$ ,  $c_2$ , and  $c_3$  are the user-defined weighting constants. We explain the terms in (5) in detail as follows.

- 1) The first term in (5) is for Objective 1). The smaller the value of it, the larger the farm's power generation.

- 2) The second term in (5) is for Objective 2). Particularly,  $F_i$  is the axial force that the wind flow exerts on  $WT_i$ , and it is defined as [8], [21]

$$F_i = \frac{1}{2} \rho A_i U_i^2 \cos^2(\beta_i) C'_{T_i}. \quad (6)$$

Here,  $\rho$  denotes the air density,  $A_i$  denotes the swept area of the rotor plane, and  $C'_{T_i}$  is called the modified thrust coefficient. Following [21], the whole term is related to dynamical turbine loading. Therefore, MPDRL can balance power generation maximization with dynamical load minimization by introducing this term into (5).

- 3) The third term is used to avoid unacceptable yaw offsets and large yaw-induced structural loads, and here  $b_\beta$  denotes the maximum acceptable yaw angle.

*Remark 1:* To test the performance of the MPDRL algorithm proposed in this brief, we use the dynamic WFSim proposed in [8] to carry out case studies (see Section IV). As explained in [8] and adopted in many studies (e.g., [7], [9], [21]), WFSim uses the modified thrust coefficient  $C'_{T_i}$  and the yaw angle  $\beta_i$  or their changes as the control variables. It is noteworthy that  $C'_{T_i}$  is directly related to the turbine set point in terms of blade pitch angle and rotational speed (see Appendix in [8] for details). Moreover, it should be emphasized that the proposed MPDRL method is data-driven, and its learning process does not rely on the underlying models of any specific WFSims. WFSim is used in this brief because it is a control-oriented simulator that can keep the key features of dynamic flow fields and wind farms while balancing simulation fidelity and computational complexity. Therefore, to indicate that other proper induction-related states can also be used as control variables in our MPDRL, we still denote the induction-related state as  $\alpha_i$  instead of directly denoting it as  $C'_{T_i}$  in MPDRL design to keep generality.

The main control goal is finding the best control policies for  $\delta\alpha$  and  $\delta\beta$  to minimize the following long-term performance metric, where  $\gamma \in (0, 1]$  is a constant discount factor:

$$V = \sum_{k=0}^{\infty} \gamma^k r(k). \quad (7)$$

We note that solving this task is challenging, and it is intractable for the conventional control methods. The main difficulties are from the following aspects.

- 1) As mentioned in the introduction, the wind-farm control tasks are commonly subject to complicated aerodynamic couplings among turbines. Particularly, due to the existence of wake effects, the power generation of downstream turbines is not only decided by the control actions of themselves but also influenced by that of other turbines. Particularly,  $w$  in (4) is related to  $\bar{x}$  in (3) and the time-varying inflow wind speed (denoted by  $U_\infty$ ). Such a complicated relationship is challenging to be accurately modeled given the stochastic nature of wake effects and  $U_\infty$ . It also renders no analytical solution for  $V$  in (7) and hinders the feasibility of the conventional optimal control methods. This brief addresses this complex task via DRL, which allows us to approximate



$V$  and learn the optimal control policies by DNNs, achieving data-driven farm-level power maximization under time-varying wind speeds without requiring any analytical models.

- 2) Wake effects have time-delayed features. The wake changes induced by control actions typically require tens/hundreds of seconds to propagate from upstream turbines to downstream turbines, leading to difficulties in evaluating the control performance from a long-term perspective. Moreover, this issue also renders the whole task to be a partially observable Markov decision process, blocking the direct application of mainstream DRL algorithms, such as DDPG [18]. To handle this issue in this brief, we use a multiplayer DRL structure to encode look-back data and evaluate the long-term reward  $V$  while considering the stochastic nature of wind speeds.

### III. DESIGN OF A MULTIPLAYER DRL ALGORITHM FOR WIND FARM CONTROL

To handle the time-delayed feature of wake effects and alleviate the non-Markovian feature of the whole control task, we feed not only the instantaneous measurements of  $x$  but also its past data into the wind farm control method. Particularly, we define

$$x(k) = [\bar{x}(k-l), \bar{x}(k-l+1), \dots, \bar{x}(k)]^T \quad (8)$$

where  $l \geq 0$  denotes a user-defined look-back step to mitigate the non-Markovian feature. Then we can describe the whole wind farm control system as follows:

$$x(k+1) = f(x(k), \delta\alpha, \delta\beta, w). \quad (9)$$

However, due to the wake effect's inherent complexity and stochastic nature,  $f$  is unknown for controller design. Based on the principle of  $H_\infty$  control technique, the control objective is to learn the optimal control policies  $\delta\alpha^*$  and  $\delta\beta^*$  under the influence of the potentially worst case  $w$  (denoted by  $w^*$ ), formalized by

$$\delta\alpha^*(k), \delta\beta^*(k) = \arg \min_{\delta\alpha, \delta\beta} V^*(k). \quad (10)$$

Here,  $V^*(k)$  is based on the following definition for any  $k \in \mathbb{N}$ :

$$V^*(k) = \min_{\delta\alpha, \delta\beta} \max_w \{V(x(k), \delta\alpha(k), \delta\beta(k), w(k))\}. \quad (11)$$

Given all these preliminaries, we can summarize the wind farm control task as follows.

Solving  $\delta\alpha^*$ ,  $\delta\beta^*$ , and  $w^*$  for

$$V^*(k) = \min_{\delta\alpha, \delta\beta} \max_w \{V(x(k), \delta\alpha(k), \delta\beta(k), w(k))\} \quad (12)$$

S.t.

$$x(k+1) = f(x(k), \delta\alpha, \delta\beta, w) \quad (13)$$

$$\delta\alpha_{\min} \leq \delta\alpha_i \leq \delta\alpha_{\max}, \quad i = 1, 2, \dots, n \quad (14)$$

$$\delta\beta_{\min} \leq \delta\beta_i \leq \delta\beta_{\max}, \quad i = 1, 2, \dots, n \quad (15)$$

$$-1 \leq \bar{\alpha}_i \leq 1, \quad i = 1, 2, \dots, n \quad (16)$$

$$-1 \leq \bar{\beta}_i \leq 1, \quad i = 1, 2, \dots, n \quad (17)$$

$$\bar{U}_i \in \mathcal{L}_\infty, \quad i = 1, 2, \dots, n. \quad (18)$$

Here,  $\delta\alpha_{\min}$ ,  $\delta\alpha_{\max}$ ,  $\delta\beta_{\min}$ , and  $\delta\beta_{\max}$  are the bounds for one-step control actions.

With (12)–(18), the wind farm control task is transformed to a game problem with two minimizing players (i.e.,  $\delta\alpha$  and  $\delta\beta$ ) and one maximizing player (i.e.,  $w$ ). However, since  $f$  is an unknown function with high nonlinearity and complexity, it is impossible to analytically solve the Nash equilibrium  $\{\delta\alpha^*, \delta\beta^*, w^*\}$  for this game. Instead, we propose a multiplayer DRL algorithm to approximate  $V^*$  and the corresponding optimal control policies  $\delta\alpha^*$  and  $\delta\beta^*$ .

An important property of the game problem considered here is that  $V^*$  satisfies the so-called discrete-time Hamilton–Jacobi–Isaacs (HJI) equation [22], which is in line with the principle of optimality

$$V^*(k) = \min_{\delta\alpha, \delta\beta} \max_w \{r(k) + \rho V^*(k+1)\} \quad \forall k \in \mathbb{N}. \quad (19)$$

After that, we define the  $Q$ -function [16], [17], [23], [24]

$$\begin{aligned} Q_{\delta\alpha, \delta\beta, w}(x(k), a_\alpha, a_\beta, d) &= \sum_{i=k+1}^{\infty} \gamma^{i-k} r(x(k+1), \delta\alpha(k+1), \delta\beta(k+1), w(k+1)) \\ &\quad + r(x(k), a_\alpha, a_\beta, d) \\ &= \gamma Q_{\delta\alpha, \delta\beta, w}(x(k+1), \delta\alpha(k+1), \delta\beta(k+1), w(k+1)) \\ &\quad + r(x(k), a_\alpha, a_\beta, d). \end{aligned} \quad (20)$$

Here,  $Q_{\delta\alpha, \delta\beta, w}$  is commonly referred to as an action-state value function [16], [17], [23], [24]. It represents the value of the performance metric obtained when the inputs  $a_\alpha$ ,  $a_\beta$ , and  $d$  are applied at time step  $k$ , and the policies  $\delta\alpha$ ,  $\delta\beta$ , and  $w$  are pursued thereafter [24]. It should be emphasized that (20) also applies to the Nash equilibrium  $\{\delta\alpha^*, \delta\beta^*, w^*\}$ , i.e.,

$$\begin{aligned} Q_{\delta\alpha^*, \delta\beta^*, w^*}(x(k), a_\alpha, a_\beta, d) &= \gamma Q_{\delta\alpha^*, \delta\beta^*, w^*}(x(k+1), \delta\alpha(k+1), \delta\beta(k+1), w(k+1)) \\ &\quad + r(x(k), a_\alpha, a_\beta, d). \end{aligned} \quad (21)$$

The  $Q$ -function defined above plays a key role in the learning of control policies and in the update of DNN parameters.

A specially designed critic–actor–distractor structure is used in our MPDRL algorithm. An illustration that shows the main modules and frameworks of MPDRL is given in Fig. 1. Particularly, the critic aims to approximate  $V^*$ ; the two actors are used to learn the optimal control policies for  $\delta\alpha$  and  $\delta\beta$  (i.e.,  $\delta\alpha^*$  and  $\delta\beta^*$ ), respectively; and the function of the distractor is to evaluate the worst case disturbance, i.e.,  $w^*$ . All these modules are built by DNNs. We denote the parameters of the critic, the two actors, and the distractor DNNs as  $\theta^Q$ ,  $\theta^\alpha$ ,  $\theta^\beta$ , and  $\theta^w$ , respectively, and their outputs are  $Q$ ,  $\delta\alpha$ ,  $\delta\beta$ , and  $w$ , respectively.

Furthermore, an additional set of DNNs is also used, which is the “soft copy” of the main critic–actor–distractor structure. Particularly, these DNNs have exactly the same neuron types and numbers and network layers as their counterparts in the main critic–actor–distractor structure. These additional DNNs form a target critic–actor–distractor structure. We denote their parameters as  $\theta^{Q'}$ ,  $\theta^{\alpha'}$ ,  $\theta^{\beta'}$ , and  $\theta^{w'}$ , respectively, and their outputs are denoted by  $Q'$ ,  $\delta\alpha'$ ,  $\delta\beta'$ , and  $w'$ , respectively. These

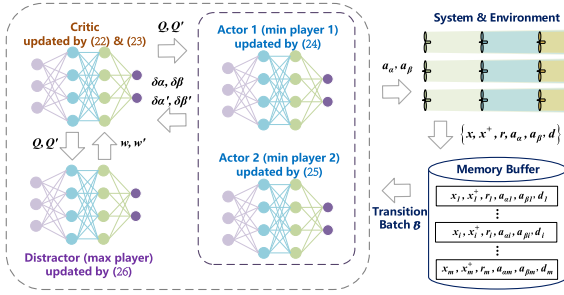


Fig. 1. Main structure of MPDRL.

parameters slowly track their counterparts with a small positive rate  $\tau$ . It should be emphasized that the idea of using such an additional set of DNNs is originally from [17]. This design can significantly improve the learning stability, which has been proven by many relevant studies [4], [14], [17], [18], [25].

Moreover, we also use the experience replay strategy [17], [18] in our MPDRL to mitigate the correlation issue of sequential learning data. Specifically, as shown in Fig. 1, a memory buffer  $\mathcal{M}$  (with a size of  $m$ ) is used to store the system data  $\{x(k), x(k+1), r(k), \delta\alpha(k), \delta\beta(k), d(k)\}$  (which is commonly referred to as a transition) at every time step  $k$ . It is noteworthy that here  $d(k)$  is the actual disturbance observed at time step  $k$  instead of the policy  $w(k)$  generated by the distractor. For each learning iteration, a small batch (with a size of  $b$ ) of transitions (denoted by  $\{x_i, x_i^+, r_i, \delta\alpha_i, \delta\beta_i, d_i\}$ ,  $i = 1, 2, \dots, b$ ) are randomly selected from  $\mathcal{M}$  to update DNNs.

Then we are ready to introduce the updating laws for the main critic-actor-distractor structure. Based on the  $Q$ -function in (20), we define the following temporal difference error (TD-error) for any transition  $\{x_i, x_i^+, r_i, \delta\alpha_i, \delta\beta_i, d_i\}$ :

$$e_i = r_i + \gamma Q'(x_i^+, \delta\alpha'(x_i^+|\theta^{\alpha'}), \delta\beta'(x_i^+|\theta^{\beta'}), w'(x_i^+|\theta^{w'})|_{\theta^{Q'}}) - Q(x_i, \delta\alpha_i, \delta\beta_i, d_i|\theta^Q). \quad (22)$$

The critic aims to minimize the amplitude of  $e_i$  for all the transitions  $i$  in a sampled batch at every learning iteration. To this end, we define the following loss function for the transition batch, which directly drives the updating of the main critic's parameters (i.e.,  $\theta^Q$ ):

$$L = \sum_{i=1}^b \|e_i\|^2. \quad (23)$$

As discussed before, the actors  $\delta\alpha$  and  $\delta\beta$  are minimizing players that aim to minimize the value of the long-term performance metric (which is estimated by  $Q$ ), while the purpose of the distractor  $w$  is the opposite. Based on that, their parameters' updating process can be driven by the corresponding gradients of  $Q$ , which are defined as follows:

$$\nabla_{\theta^{\alpha}} = \frac{1}{b} \sum_{i=1}^b [\nabla_{\delta\alpha} Q(x_i, \delta\alpha, \delta\beta, w) \cdot \nabla_{\theta^{\alpha}} \delta\alpha(x_i|\theta^{\alpha})] \quad (24)$$

$$\nabla_{\theta^{\beta}} = \frac{1}{b} \sum_{i=1}^b [\nabla_{\delta\beta} Q(x_i, \delta\alpha, \delta\beta, w) \cdot \nabla_{\theta^{\beta}} \delta\beta(x_i|\theta^{\beta})] \quad (25)$$

$$\nabla_{\theta^w} = \frac{1}{b} \sum_{i=1}^b [\nabla_w Q(x_i, \delta\alpha, \delta\beta, w) \cdot \nabla_{\theta^w} w(x_i|\theta^w)]. \quad (26)$$

In every learning iteration, the updating process of  $\theta^{\alpha}$  and  $\theta^{\beta}$  is driven by  $-\nabla_{\theta^{\alpha}}$  and  $-\nabla_{\theta^{\beta}}$ , respectively, while  $\theta^w$  is updated by  $\nabla_{\theta^w}$ .

Based on all these designs, we summarize our MPDRL algorithm for wind farm control in Algorithm 1.

**Remark 2:** The proposed MPDRL algorithm is built upon the  $Q$ -learning theory [16], [23], [24]. One can refer to [23] for the key idea behind a standard  $Q$ -learning algorithm and its general convergence analysis. Distinct from the mainstream DRL algorithms that also use  $Q$ -learning or its variants such as Deep  $Q$ -Network [17] and Deep Deterministic Policy Gradient [18], we propose a special multiplayer structure consisting of two actors (i.e.,  $\delta\alpha$  and  $\delta\beta$ ) that aim to minimize the reward function and a distractor (i.e.,  $w$ ) that has the opposite objective. On one hand, this design can evaluate the potential worst case disturbances, guiding control policies and enhancing the whole system's robustness. On the other hand, using two separated actors (instead of aggregating the yaw and induction control signals together) can adapt to distinctive features (e.g., different changing rates) of different control inputs and enhance the learning effectiveness and the algorithm's overall performance.

#### IV. CASE STUDIES

In this section, we use WFSim [8] to test the performance of MPDRL. As mentioned in Remark 1, WFSim uses the modified thrust coefficient (i.e.,  $C_{T_i}'$ ) and yaw angles or their changes as control variables. Therefore, we replace all the terms related to  $\alpha_i$  in MPDRL with the corresponding terms in  $C_{T_i}'$  to adapt to the requirement in WFSim.

In addition to the MPDRL method proposed in this brief, we also use three other wind farm control methods in simulations to compare their performance with our MPDRL.

- 1) *Greedy Control Strategy*: This strategy is the benchmark for the wind farm control tasks, and it is currently the most commonly employed wind farm control strategy in the industry. In the greedy strategy, every turbine in the farm aims to maximize its own power generation without considering the influence of other turbines. Following that and [8], one can set  $C_{T_i}' \equiv 2, \beta_i \equiv 0^\circ, i = 1, 2, \dots, n$ , for all the turbines in the farm to conduct the greedy strategy.
- 2) *Model-Based Wind Farm Control Method (MB-WFC) in [3]*: This important method carries out receding-horizon optimization based on the FLORIS model. It can achieve closed-loop wind farm control under time-varying wind conditions—a very suitable candidate to compare with MPDRL.
- 3) *DRL-Based Method*: This method has the same main structure and settings as MPDRL, but it does not have the distractor. We mention it as DRL-W/D in simulations. It can help us test the robustness of MPDRL further and show the advantage of the multiplayer structure.

Two simulation scenarios are considered in case studies. Specifically, we use a prototypical wind farm with nine NREL

**Algorithm 1** Multiplayer DRL (MPDRL) Algorithm for Wind Farm Control

- Aggregate and normalize the system states, control inputs and disturbances to transform the wind farm control problem into a multi-player game as described in Eqs. (12)-(18).
- Decide hyperparameters for the critic-actor-distractor DRL structure, including networks' layer numbers and neuron numbers & types for each layer.
- Initialize  $\theta^Q$ ,  $\theta^\alpha$ ,  $\theta^\beta$ ,  $\theta^w$ ,  $\theta^{Q'}$ ,  $\theta^{\alpha'}$ ,  $\theta^{\beta'}$ ,  $\theta^{w'}$  and all the other user-defined parameters.

```

1: for each learning episode do
2:   for  $k = 0$  to the maximum steps per episode do
3:     Based on the current system state  $x(k)$ , generate control actions  $a_\alpha$  and  $a_\beta$  via  $a_\alpha = \delta\alpha(x(k)|\theta^\alpha) + \epsilon_\alpha(k)$  and  $a_\beta = \delta\beta(x(k)|\theta^\beta) + \epsilon_\beta(k)$ , respectively, where  $\epsilon_\alpha(k)$  and  $\epsilon_\beta(k)$  are noises for exploration purposes.
4:     Apply the control actions  $a_\alpha$  and  $a_\beta$  to the wind farm control system, and observe  $x(k+1)$ ,  $d$ , and  $r$ .
5:     Organize the transition  $\{x(k), x(k+1), r, a_\alpha, a_\beta, d\}$  and store it in the memory buffer  $\mathcal{M}$ .
6:     Sample a batch of transitions from  $\mathcal{M}$ , denoted by  $\{x_i, x_i^+, r_i, a_{\alpha i}, a_{\beta i}, d_i\}$ ,  $i = 1, \dots, n$ .
7:     Update the critic's parameter  $\theta^Q$  via the loss function  $L$  defined by Eqs. (22) and (23).
8:     Update  $\theta^\alpha$ ,  $\theta^\beta$  and  $\theta^w$  via Eqs. (24) - (26).
9:     Update  $\theta^{Q'}$ ,  $\theta^{\alpha'}$ ,  $\theta^{\beta'}$  and  $\theta^{w'}$  via 'soft replacement'.
10:  end for
11: end for

```

TABLE I  
SIMULATION SETTINGS

Parameters	Values
$C'_{T\min}, C'_{T\max}, \beta_{\min}, \beta_{\max}, b_\beta$	0.1, 2, 30°, 30°, 30°
$\delta C'_{T\min}, \delta C'_{T\max}, \delta\beta_{\min}, \delta\beta_{\max}$	0.1, 0.1, 1°, 1°
$c_1, c_2, c_3$	1, 0.05, 0.1
$l, \gamma, \tau, b, m$	100, 0.99, 0.05, 128, 10000

5-MW wind turbines in the first scenario. Then a large-scale wind farm from the European CL-Windcon<sup>1</sup> project that contains 80-DTU 10-MW wind turbines is used to test the scalability of MPDRL.

#### A. Case Study With a Prototypical Nine-Turbine Wind Farm

In this case study, we simulate a  $2518.8 \times 1558.4$  m flow field with a wind farm that consists of nine NREL 5-MW turbines in WFSim. A bird's-eye view of the flow field and the wind farm is given in Fig. 2. Our method needs to use DNNs that can handle time-series data, such as LSTM networks, GRU, and transformer networks. Without loss of generality, here we use LSTM DNNs in our case studies. We carry out DNN training based on the settings in Table I. It is noteworthy that time-varying wind conditions are used in training and testing of MPDRL. A 700-s example for a

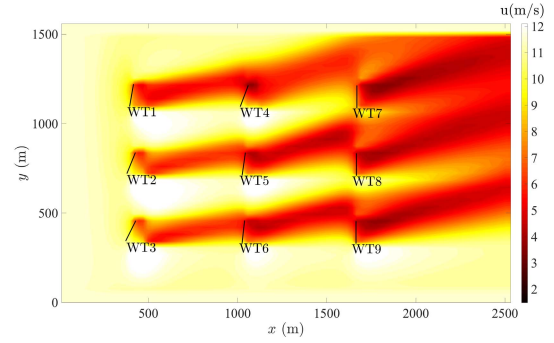


Fig. 2. Flow field under MPDRL (at  $t = 700$  s).

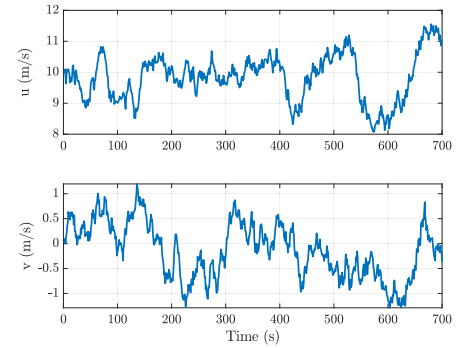


Fig. 3. Wind speed ( $u$ : longitudinal;  $v$ : lateral).

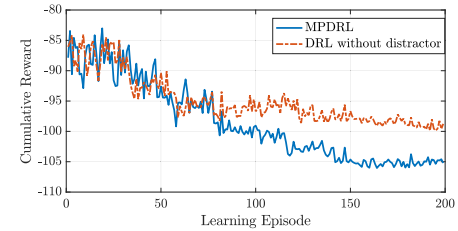


Fig. 4. Cumulative reward in DRL learning.

time-varying wind profile is given in Fig. 3, with  $u$  being the longitudinal wind speed and  $v$  being the lateral wind speed. It should be emphasized that the wind profile differs in each learning episode, and our MPDRL can only use current and past measured wind conditions (no preview information for the future). In addition, measurement errors and process noises are also taken into account. Specifically, zero-mean Gaussian noises with standard deviations 0.5 and 0.05 are set to be the measurement errors of  $\beta_i$  and  $C'_{T_i}$ , respectively; the control signals  $\delta\beta_i$  and  $\delta C'_{T_i}$  are also polluted by such noises but the standard deviations are 0.1 and 0.01, respectively.

Given all these settings, the learning curves (i.e., the curve of the cumulative reward  $r$  per learning episode with 200 steps) of the two DRL-based wind farm control methods are shown in Fig. 4. Though both the methods have the essential learning ability to improve the control policy (we aim to minimize rewards in this brief), MPDRL leads to clearly superior learning performance compared with the case without the distractor module. It has an averagely lower episode reward level under the influence of the same exogenous inputs, measurement errors, and process noises (though these disturbances differ in each episode).

<sup>1</sup><http://www.clwindcon.eu/>



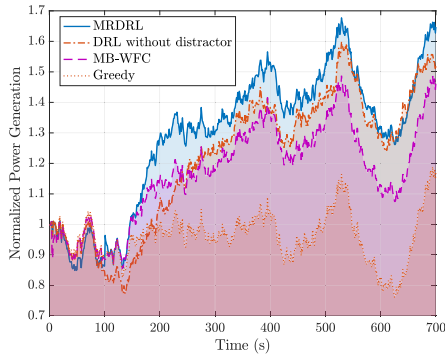


Fig. 5. Normalized farm-level power generation.

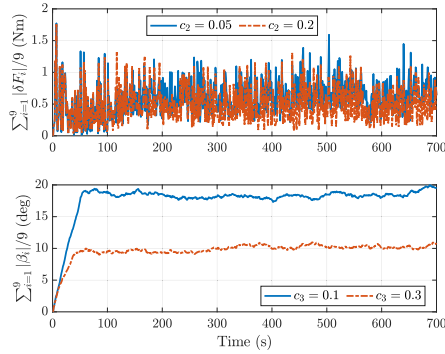


Fig. 6. Responses of load-related terms.

Based on the training results, we test MPDRL by a 700-s run with the wind profile shown in Fig. 3 and initial conditions as  $C_{Ti}(0) = 2, \beta_i(0) = 0^\circ, i = 1, 2, \dots, n$ . The simulations with the other three strategies are also conducted for performance validation and comparison purposes. The flow field at  $t = 700$  s under MPDRL is shown in Fig. 2. One can see that MPDRL successfully achieves wake steering. The normalized power outputs (w.r.t. the greedy power output at  $t = 0$  s) under different controllers are illustrated in Fig. 5. It can be observed that MPDRL, DRL-W/D, and MB-WFC all lead to power increases with respect to the greedy strategy, and MPDRL has the best performance among all these methods—MPDRL achieves a 34.23% power increase on average compared with the greedy strategy during the 700-s period, while that of DRL-W/D and MB-WFC are 26.27% and 21.78%, respectively.

These results demonstrate the robustness of MPDRL from two aspects: 1) compared with the important model-based method in [3], MPDRL is data-driven and model-free, showing robustness to potential uncertainties and modeling errors and leading to higher power generation in simulations and 2) MPDRL also leads to better learning and testing performance than DRL-W/D in the case study. As mentioned before, the key difference between them is the distractor module in our multiplayer structure. As verified by simulations, that brings MPDRL the robustness against the exogenous disturbance  $w$ .

As discussed in Section II, MPDRL should balance two competing objectives: 1) maximizing the farm-level power generation (core objective) and 2) avoiding large loads induced by induction and yaw control (secondary objective). The trade-off between these two objectives can be achieved by

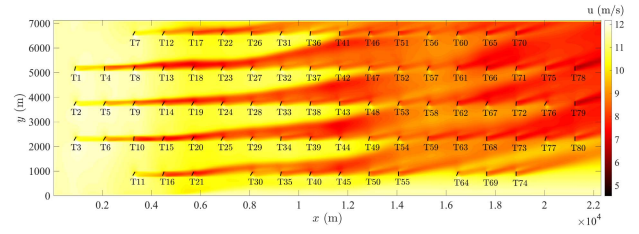


Fig. 7. Flow field of the 80-turbine farm under MPDRL.

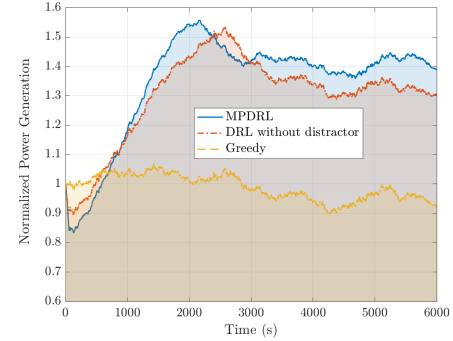


Fig. 8. Normalized power of the 80-turbine wind farm.

adjusting the parameters  $c_1$ ,  $c_2$ , and  $c_3$  in the definition of  $r$  in (5). To illustrate that, we conduct two additional simulations for MPDRL under different parameters: 1) adjust  $c_2$  from 0.05 to 0.2 and keep all the other parameters unchanged. That enlarges the weight of the term  $(c_2/n) \sum_{i=1}^n |F_i(k) - F_i(k-1)|$  in (5). As mentioned in Section II, this term is related to the dynamical turbine loading and 2) adjust  $c_3$  from 0.1 to 0.3 and keep all the other parameters unchanged. That enlarges the weight of the term  $(c_3/n) \sum_{i=1}^n |(\beta_i(k)/b_\beta)|$ , aiming to avoid unacceptable yaw offsets and large yaw-induced structural loads. The simulation results of MPDRL under these changes are given in Fig. 6. Compared with the results with the original settings, the averaged dynamical turbine load and the averaged yaw offset are reduced by 19.49% and 45.34%, respectively.

### B. Case Study With a Large-Scale Wind Farm

To test the scalability of MPDRL, we consider a large wind farm with 80-DTU wind turbines (layout illustrated in Fig. 7) in this case study. Notably, the flow field in this case study is 40 times larger than the one in Section IV-A. Therefore, a 6000-s run under stochastic wind speeds is used to comprehensively test the performance of MPDRL from a long-term standpoint. The flow field at  $t = 6000$  s under MPDRL is given in Fig. 7. One can see that the wake effect in this case study is much more complicated than in Section IV-A. The normalized power outputs (with respect to the greedy power generation at  $t = 0$  s) under different control strategies are illustrated in Fig. 8. It indicates that MPDRL can still lead to a significant farm-level power generation increase compared with the benchmark—an over 34.3% increase in average is achieved. Moreover, its performance is better than DRL-W/D, showing its strong robustness.

## V. CONCLUSION

A DRL-based wind farm control method was proposed in this brief to maximize the farm-level power generation under

strong wake effects and stochastic wind speeds. Distinct from the conventional wind farm control methods, the proposed method is data-driven and model-free—it does not require any analytical models (e.g., wake model) to carry out wind farm control. Benefiting from the multiplayer concept, the robust control technique, and a specially designed critic–actor–distractor structure, our DRL-based method has good robustness, adaptability, and applicability. The simulation results verified the proposed method’s effectiveness. It increased farm-level power generation by over 30% compared with the greedy strategy and showed better performance than relevant approaches. It should be emphasized that the rate of farm-level power generation improvement can be influenced by many factors, including the operating conditions (e.g., the main wind direction), the wind farm layouts (e.g., the longitudinal and lateral distances between turbines), and also wind turbine types. The simulation results demonstrated were for the wind farms with the operating conditions and specifications in case studies. Particularly, the dominant wind direction was along the rows of wind turbines. Though this is a commonly used and most adopted setting in relevant studies, it could not represent the entire lifetime operating conditions of wind farms. Therefore, the magnitudes of benefit in case studies could not be achieved when considering the full-lifetime operations of real wind farms. It is noteworthy that though different operation conditions and specifications vary the magnitude of benefit, our DRL algorithm always aims to learn the optimal wind farm control strategies, and it can adapt to different scenarios. As future work, hybrid reinforcement learning and grouping strategies will be explored in the future to enhance the learning efficiency and reduce the computational complexity of the proposed method.

## REFERENCES

- [1] *Global Wind Report*, GWEC, Brussels, Belgium, 2021.
- [2] P. M. O. Gebraad et al., “Wind plant power optimization through yaw control using a parametric model for wake effects—A CFD simulation study,” *Wind Energy*, vol. 19, no. 1, pp. 95–114, 2016.
- [3] B. M. Doekemeijer, D. van der Hoek, and J.-W. van Wingerden, “Closed-loop model-based wind farm control using FLORIS under time-varying inflow conditions,” *Renew. Energy*, vol. 156, pp. 719–730, Aug. 2020.
- [4] H. Dong and X. Zhao, “Wind-farm power tracking via preview-based robust reinforcement learning,” *IEEE Trans. Ind. Informat.*, vol. 18, no. 3, pp. 1706–1715, Mar. 2022.
- [5] J. Annoni, P. Seiler, K. Johnson, P. Fleming, and P. Gebraad, “Evaluating wake models for wind farm control,” in *Proc. Amer. Control Conf.*, Jun. 2014, pp. 2517–2523.
- [6] T. Knudsen, T. Bak, and M. Svenstrup, “Survey of wind farm control-power and fatigue optimization,” *Wind Energy*, vol. 18, no. 8, pp. 1333–1351, Aug. 2015.
- [7] M. Vali, V. Petrović, S. Boersma, J.-W. van Wingerden, L. Y. Pao, and M. Kühn, “Adjoint-based model predictive control for optimal energy extraction in waked wind farms,” *Control Eng. Pract.*, vol. 84, pp. 48–62, Mar. 2019.
- [8] S. Boersma, B. Doekemeijer, M. Vali, and J.-W. V. Wingerden, “A control-oriented dynamic wind farm model: WFSim,” *Wind Energy Sci.*, vol. 3, no. 1, pp. 75–95, 2018.
- [9] K. Chen, J. Lin, Y. Qiu, F. Liu, and Y. Song, “Model predictive control for wind farm power tracking with deep learning-based reduced order modeling,” *IEEE Trans. Ind. Informat.*, vol. 18, no. 11, pp. 7484–7493, Nov. 2022.
- [10] J. R. Marden, S. D. Ruben, and L. Y. Pao, “A model-free approach to wind farm control using game theoretic methods,” *IEEE Trans. Control Syst. Technol.*, vol. 21, no. 4, pp. 1207–1214, Jul. 2013.
- [11] J. Park and K. H. Law, “Bayesian ascent: A data-driven optimization scheme for real-time control with application to wind farm power maximization,” *IEEE Trans. Control Syst. Technol.*, vol. 24, no. 5, pp. 1655–1668, Sep. 2016.
- [12] H. Zhao, J. Zhao, J. Qiu, G. Liang, and Z. Y. Dong, “Cooperative wind farm control with deep reinforcement learning and knowledge-assisted learning,” *IEEE Trans. Ind. Informat.*, vol. 16, no. 11, pp. 6912–6921, Nov. 2020.
- [13] P. Stanfel, K. Johnson, C. J. Bay, and J. King, “A distributed reinforcement learning yaw control approach for wind farm energy capture maximization,” in *Proc. Amer. Control Conf. (ACC)*, Jul. 2020, pp. 4065–4070.
- [14] H. Dong, J. Zhang, and X. Zhao, “Intelligent wind farm control via deep reinforcement learning and high-fidelity simulations,” *Appl. Energy*, vol. 292, Jun. 2021, Art. no. 116928.
- [15] H. Dong and X. Zhao, “Composite experience replay-based deep reinforcement learning with application in wind farm control,” *IEEE Trans. Control Syst. Technol.*, vol. 30, no. 3, pp. 1281–1295, May 2022.
- [16] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.
- [17] V. Mnih et al., “Human-level control through deep reinforcement learning,” *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.
- [18] T. P. Lillicrap et al., “Continuous control with deep reinforcement learning,” 2015, *arXiv:1509.02971*.
- [19] M. Vali, V. Petrović, S. Boersma, J.-W. van Wingerden, and M. Kühn, “Adjoint-based model predictive control of wind farms: Beyond the quasi steady-state power maximization,” *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 4510–4515, 2017.
- [20] J. Park and K. H. Law, “A data-driven, cooperative wind farm control to maximize the total power production,” *Appl. Energy*, vol. 165, pp. 151–165, Mar. 2016.
- [21] S. Boersma, B. M. Doekemeijer, S. Siniscalchi-Minna, and J. W. van Wingerden, “A constrained wind farm controller providing secondary frequency regulation: An LES study,” *Renew. Energy*, vol. 134, pp. 639–652, Apr. 2019.
- [22] H. Zhang, C. Qin, B. Jiang, and Y. Luo, “Online adaptive policy learning algorithm for  $H_\infty$  state feedback control of unknown affine nonlinear discrete-time systems,” *IEEE Trans. Cybern.*, vol. 44, no. 12, pp. 2706–2718, Dec. 2014.
- [23] 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.
- [24] B. Luo, Y. Yang, and D. Liu, “Policy iteration Q-learning for data-based two-player zero-sum game of linear discrete-time systems,” *IEEE Trans. Cybern.*, vol. 51, no. 7, pp. 3630–3640, Jul. 2021.
- [25] D. Silver et al., “A general reinforcement learning algorithm that masters chess, shogi, and go through self-play,” *Science*, vol. 362, no. 6419, pp. 1140–1144, Dec. 2018.