Fusion of Microgrid Control With Model-Free Reinforcement Learning: Review and Vision

Buxin She, Student Member, IEEE, Fangxing Li, Fellow, IEEE, Hantao Cui, Senior Member, IEEE, Jingqiu Zhang, Student Member, IEEE, and Rui Bo, Senior Member, IEEE

Abstract—Challenges and opportunities coexist in microgrids as a result of emerging large-scale distributed energy resources (DERs) and advanced control techniques. In this paper, a comprehensive review of microgrid control is presented with its fusion of model-free reinforcement learning (MFRL). A high-level research map of microgrid control is developed from six distinct perspectives, followed by bottom-level modularized control blocks illustrating the configurations of grid-following (GFL) and grid-forming (GFM) inverters. Then, mainstream MFRL algorithms are introduced with an explanation of how MFRL can be integrated into the existing control framework. Next, the application guideline of MFRL is summarized with a discussion of three fusing approaches, i.e., model identification and parameter tuning, supplementary signal generation, and controller substitution, with the existing control framework. Finally, the fundamental challenges associated with adopting MFRL in microgrid control and corresponding insights for addressing these concerns are fully discussed.

Index Terms—Microgrid control, data-driven control, model-free reinforcement learning, grid-following and grid-forming inverters, review and vision.

I. INTRODUCTION

Microgrids are gaining popularity due to their capability for accommodating distributed energy resources (DERs) and form a self-sufficient system [1]. Microgrids not only contribute to the development of a zero-carbon city but also work as a fundamental component of the ‘source, network, and load’ integrated energy systems. A microgrid may incorporate various types of energy sources and act as an energy router [2], making it possible for the grid to survive severe events while also making the country more energy-resilient and secure [3].

A typical microgrid is composed of various DERs, energy storage systems, and loads that are connected locally as a united controlled entity [4]. In comparison to a traditional synchronous generator-dominated bulk power system, microgrids have a larger penetration of DERs [5], [6], a smaller system size [7], a greater degree of uncertainty [8], lower system inertia [9], [10], and a stronger coupling of voltage and frequency (V-f). All these features pose challenges to the design of a microgrid control system. A complete microgrid control system is comprised of software and hardware that can both perform microgrid functionalities and guarantee stability at the same time [11]. The software is also referred to as microgrid controllers, and focuses on control algorithm design in the paper. Existing microgrid controllers are usually designed under a hierarchal framework that includes the primary, secondary, and tertiary controllers [12], Ref. [13] conducted a thorough review of the hierarchal control of microgrids. There are also some articles providing an overview from the different perspectives of communication interfaces [14], operation modes [15], and control techniques [16]. All these reviews provided an excellent summary and future directions of microgrid control research. As a result, we synthesize the valuable viewpoints and develops a high-level research map of microgrid control based on existing work. Furthermore, modularized control blocks have been developed to dive into the design of the fundamental units of microgrids: grid-following (GFL) and grid-forming (GFM) inverters [17], which is advantageous for microgrid researchers.

Model-free controllers have been used previously in microgrid control because they are easy to set up and independent of the physical model of the microgrid components. For example, fuzzy logic controllers [18], [19] and adaptive controllers [20], [21] can adjust their output based on predefined membership functions and adaption laws, respectively. However, they are difficult to scale up and cannot deal with emerging uncertainties in microgrids. Neural network control [22], [23] is another type of well-known model-free method. Although neural network is good at perception and decision-making based on historical data, it lacks exploration capability and cannot adapt to the rapidly changing microgrid environment. Apart from the above-mentioned model-free techniques, reinforcement learning (RL) is a prominent approach that is concerned with how an intelligent agent learns to solve Markov Decision Processes (MDP) in an environment. If we do not assume knowledge or an exact mathematical model of the
environment, RL is referred to as model-free reinforcement learning (MFRL). Then, the RL agent finds the optimal policy through repeated interactions with the environment [24], [25]. MFRL is a promising data-driven and model-free approach since it is not dependent on an accurate system model and does not need as many labeled datasets as supervised learning. In addition, it has strong exploration capability and can achieve autonomous operation once set up. MFRL is gaining more and more attention due to its successful applications in video games [26], autonomous driving [27], robotics [28], and power systems [29]. Recently, researchers from DeepMind and École Polytechnique Fédérale de Lausanne developed a nonlinear, high-dimensional, and RL-based magnetic controller for nuclear fusion [30] and published their work in Nature. This indicates the great potential of implementing MFRL in engineering control (microgrid control).

For now, MFRL is still under development and needs further study. While some research has been conducted on MFRL for its application in microgrid control, there has been no in-depth review of how MFRL can be integrated into the current microgrid control framework. Hence, this paper performs a comprehensive review of the control framework of microgrids and summarizes how MFRL fuses with the existing control schemes.

Compared with other review papers on microgrid control, the main merits of this manuscript include:

- Plotting of a high-level research map of microgrid control from the perspective of operation mode, function grouping, timescale, hierarchical structure, communication interface, and control techniques.
- Development of modularized control blocks to dive into the fundamental units of microgrids: GFL and GFM inverters.
- Introduction of the mainstream MFRL algorithms and summary of MFRL application guidelines, and the answering of two important questions: i) ‘What kinds of tasks is MFRL suitable for?’; ii) ‘How can MFRL be fused with the existing microgrid control framework?’
- Discussion of the primary challenges associated with adopting MFRL in microgrid control and providing insights for addressing these concerns.

The rest of this paper is organized as follows. Section II introduces the current microgrid control framework, including a high-level research map and modularized control blocks. Section III gives a brief introduction to RL and the mainstream algorithms of MFRL. The characteristics of each algorithm and its application scenarios in microgrid control are also summarized. A full discussion of the fusion of microgrid control with MFRL is presented in Section IV, along with the associated challenges and insights. Section V concludes this paper.

II. MICROGRID CONTROL FRAMEWORK

This section first plots a high-level research map of microgrid control, and then develops modularized control blocks to dive into GFL and GFM inverters.

A. High-Level Research Map of Microgrid Control

Fig. 1 shows the high-level research map of microgrid control from the perspectives of 1) operation mode, 2) function grouping, 3) timescale, 4) hierarchical structure, 5) communication interface, and 6) control techniques. For each perspective, there are articles providing comprehensive reviews. They are denoted in Fig. 1 for the reader’s reference.

1) Operation Mode: A microgrid can operate in either grid-connected (GC) mode or islanded (IS) mode depending on its connectivity to the main grid [31], [32]. In GC mode, the microgrid keeps tracking the phase of the main grid through the phase-locking loop (PLL), and exchanges the mismatched power at the point of common coupling (PCC). In IS mode, the microgrid forms a self-sufficient system based on the local generations. Ref. [33] summarized the strategies for the seamless transition between GC and IS modes.

2) Function Grouping: To meet the objectives of the microgrid operation, the 2nd viewpoint is associated with function grouping, which specifically include the microgrid controller and device controller [34]. Grid-level controllers focus on supervisory control functions and grid interactive control functions, and they are more likely to be software-based and applied to the hardware; while device controllers focus on device-level control functions and local-area control functions, and they are more likely to be applied directly on the hardware (devices and assets).

3) Timescale: The time scale of microgrid control is tightly related with the control structure. So, it will be discussed in detail in the next discussion about hierarchical structure.

4) Hierarchical Structure: The hierarchical control structure is another specific function grouping perspective that clearly sets up the control targets for all the controllers, with which each level controller can work independently within the distinct timescales [11].

The primary controller is responsible for voltage and current control of inverters and automatic power sharing among generations while maintaining V-f stability on a timescale of seconds [35]. The indirect current control is used in the early stages [36], [37], and is later replaced by the direct current control due to its fast response and accurate current control capability [38]. More details can be found in the review paper [39]. Because the primary controller pertains to fast control actions, it predominantly determines the stability of microgrids [2]. Ref. [40] gave an overview of the primary control of microgrids. The secondary controller mitigates the V-f deviation unsolved by the primary controller in the timescale of seconds to minutes. It improves the power quality by generating supplementary signals based on the errors between the measurements and reference values. Ref. [41], [42] performed a review on the secondary control of ac microgrids. The tertiary controller mainly focuses on economic and resilient operations in the timescale of minutes to hours. It adjusts the setting points of the primary and secondary controllers by solving optimal power flow and considering the load side demand response. Some reviews can be found in [43], [44].

5) Communication Interface: Depending on the communication interface, the control structure of the microgrid can also be categorized into centralized control, decentralized control, and distributed control [45].

In centralized control, the microgrid control center coordinates the load and generation and responds to all disturbances.
It collects and processes all the local information before sending the control signals to each device. The centralized control has the advantage of accurate power-sharing and good transient performance but suffers from the high cost of the communication device and single point failure. In distributed control, each node communicates only with its adjunct nodes. Average-based, consensus-based, and event-triggered distributed algorithms are employed in microgrid control [46]. Distributed control algorithms require the connected communication graph of microgrids. They also have a reduced convergence speed as the network grows [47]. In decentralized control, the control signals are generated based on local measurements. It has the advantage of the plug-and-play capability and is free of communication channel time delay, but it suffers from inaccurate power-sharing and large V-f deviation after disturbances. Ref. [47] conducted a review from the perspective of communication interfaces and summarized some tricks to address their flaws.

6) **Control Techniques:** Both model-based and data-driven control techniques have been utilized in microgrid control. Beginning with the classical linear control theory, advanced model-based control approaches such as non-linear control, optimum control, and model-predictive control (MPC) are then extensively used in microgrids. Ref. [48] summarized the advances and opportunities of employing MPC in microgrids, and [49] reviewed the robust control strategies in microgrids. To address the problems of model uncertainty and unavailability, a variety of data-driven methodologies such as cutting-edge machine learning (ML) and deep learning (DL) are also employed in microgrid control. Ref. [50] reviewed the application of big data in microgrids, and [51] conducted a survey on DL for microgrid load and DER forecasting. A review of MFRL for microgrid control has yet to be done, which is why it is the main scope of this manuscript.

In summary, MFRL is a promising approach that is worth investigating and being employed in microgrids. As shown in the high-level research map, MFRL doesn’t mean to replace the existing control framework, but to complement it, improve it in a data-driven way, and finally work as an integrated part of the microgrid controller.

### B. Configuration of Grid-Following and Grid-Forming Inverters

GFL and GFM inverters are no doubt one of the most important units in microgrids [52]. This subsection develops the modularized control blocks to present the bottom-level control details of GFL and GFM inverters. Fig. 2 shows the diagram of the modularized control blocks, with which a GFL or GFM inverter can be configured easily by connecting the modules in series. In addition, it is beneficial to the fusion summary in Section IV because the diagram clearly shows the control details that could couple with MFRL.

1) **M1: Grid ∪ Inverter Module:** The 1st module (M1) is named the ‘Grid ∪ Inverter Module’ because it illustrates the
connection of an inverter to the main grid. As shown in Fig. 2, the dc source, dc-ac inverter, and RLC filter are linked in series, which are then connected to the main grid through the PCC. Here, an average model of an inverter that neglects the switching of pulse-width modulation (PWM) is often employed for the control system design. All the high-level controllers work together to generate the reference terminal voltage \( e_{abc}^{\text{ref}} \) for PWM.

2) M2: Terminal Voltage-Ref Module: The 2nd module (M2) is named the ‘Terminal Voltage-ref Module’ since it directly generates the reference terminal voltage. The control model is formulated using Kirchhoff’s current law (KCL) from \( e_{abc} \) to \( u_{abc} \) and conducting Park transformation. Then, after implementing proportional-integral (PI) controllers, the physical model and control transfer function in \( dq \) framework are shown in (1) and (2), respectively.

\[
L_d \frac{d i_d}{dt} + w L_f [-i_q] = \begin{bmatrix} e_d \\ e_q \end{bmatrix} - \begin{bmatrix} u_d \\ u_q \end{bmatrix} \\
\begin{bmatrix} e_d \\ e_q \end{bmatrix} = \begin{bmatrix} u_d \\ u_q \end{bmatrix} + w L_f [-i_q] \\
+ \begin{bmatrix} k_{pI} + k_{ud} & 0 \\ 0 & k_{pI} + k_{iI} \end{bmatrix} \left( \begin{bmatrix} i_{d}^{\text{ref}} \\ i_{q}^{\text{ref}} \end{bmatrix} - \begin{bmatrix} i_d \\ i_q \end{bmatrix} \right)
\]

(1)

\[
L_f \frac{d i_q}{dt} + \frac{w L_f}{\omega} [-i_d] = \begin{bmatrix} k_{pI} + k_{ud} & 0 \\ 0 & k_{pI} + k_{iI} \end{bmatrix} \left( \begin{bmatrix} i_{d}^{\text{ref}} \\ i_{q}^{\text{ref}} \end{bmatrix} - \begin{bmatrix} i_d \\ i_q \end{bmatrix} \right)
\]

(2)

3) M3: Current-Ref Module: The 3rd module (M3) is named the ‘Current-ref Module’ since it generates the reference current \([i_{d}^{\text{ref}}, i_{q}^{\text{ref}}]\) for M2. For a GFL inverter, \([i_{d}^{\text{ref}}, i_{q}^{\text{ref}}]\) are regulated based on the error between the actual output and the reference value. Eqs. (3)-(4) show the transfer function of M3 using PI controllers, where two low-pass filters are used to filter measured power output.

\[
\begin{bmatrix} i_{d}^{\text{ref}} \\ i_{q}^{\text{ref}} \end{bmatrix} = \begin{bmatrix} k_{pP} + k_{ud} & 0 \\ 0 & k_{pQ} + k_{iQ} \end{bmatrix} \begin{bmatrix} P - P_{\text{ref}} \\ Q - Q_{\text{ref}} \end{bmatrix}
\]

(3)

\[
P = \begin{bmatrix} \frac{1}{T_{pP}+1} & 0 \\ 0 & \frac{1}{T_{pQ}+1} \end{bmatrix} \begin{bmatrix} P_{\text{mes}} \\ Q_{\text{mes}} \end{bmatrix}
\]

(4)

For a GFM inverter, its physical model is formulated using Kirchhoff’s voltage law (KVL) at point \( u_{abc} \). After Park transformation and PI controller integration, the algebraic equation and control transfer function in \( dq \) framework are shown in (5) and (6), respectively.

\[
C_f \frac{dx}{dt} + w C_f [-u_q] = \begin{bmatrix} i_d \\ i_{q} \end{bmatrix} - \begin{bmatrix} i_{d}^{\text{ref}} \\ i_{q}^{\text{ref}} \end{bmatrix}
\]

(5)

\[
\begin{bmatrix} i_{d}^{\text{ref}} \\ i_{q}^{\text{ref}} \end{bmatrix} = \begin{bmatrix} i_{d} \\ i_{q} \end{bmatrix} + w C_f [-u_q] \\
+ \begin{bmatrix} k_{pud} + k_{iud} & 0 \\ 0 & k_{puq} + k_{iug} \end{bmatrix} \left( \begin{bmatrix} u_{d}^{\text{ref}} \\ u_{q}^{\text{ref}} \end{bmatrix} - \begin{bmatrix} u_d \\ u_q \end{bmatrix} \right)
\]

(6)
4) M4: Power ∩ Voltage Module: The 4th module (M4) is named the ‘Power ∩ Voltage Module’ which indicates the fundamental difference between GFL and GFM inverters. A GFL inverter is controlled as a current source and requires a power reference as an input, while a GFM inverter is controlled as a voltage source and needs a voltage reference as an input [39]. Another big difference is that a GFL inverter needs a PLL to track the phase of the main grid while a GFM inverter is self-synchronized [53]. Droop control is the most widely used control method in microgrids. It takes advantage of the coupling between power generation and the grid V-f [54]. Typically, an inductive microgrid employs the $P-f$ and $Q-V$ droop curves, while resistive microgrids use the reverse $P-V$ and $Q-f$ droop curves. The M4 plotted in Fig. 2 shows the control blocks for an inductive microgrid, and their control models are shown below.

- Droop-controlled GFL inverter

$$\begin{bmatrix} P_{ref} \\ Q_{ref} \end{bmatrix} = \begin{bmatrix} k_f & 0 \\ 0 & k_v \end{bmatrix} \begin{bmatrix} w_0 \\ U_0 \end{bmatrix} - \begin{bmatrix} w_{mes} \\ U_{mes} \end{bmatrix} + \begin{bmatrix} P_0 \\ Q_0 \end{bmatrix} \tag{7}$$

- Droop-controlled GFM inverter

$$\begin{bmatrix} w_{ref} \\ u_{dref} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} P_0 \\ Q_0 \end{bmatrix} - \begin{bmatrix} P_{mes} \\ Q_{mes} \end{bmatrix} + \begin{bmatrix} w_0 \\ V_0 \end{bmatrix} \tag{8}$$

To provide more inertia support to microgrids leveraging DERs, the virtual synchronous generator (VSG) control method is proposed to emulate the behavior of synchronous generators [55]. Mathematically speaking, the VSG belongs to proportional-differential control. Below is the transfer function of the GFL and GFM inverters implementing the VSG.

- VSG-controlled GFL inverter

$$\begin{bmatrix} P_{ref} \\ Q_{ref} \end{bmatrix} = \begin{bmatrix} D + Ks & 0 \\ 0 & k_v \end{bmatrix} \begin{bmatrix} w_0 \\ U_0 \end{bmatrix} - \begin{bmatrix} w_{mes} \\ U_{mes} \end{bmatrix} + \begin{bmatrix} P_0 \\ Q_0 \end{bmatrix} \tag{9}$$

- VSG-controlled GFM inverter

$$\begin{bmatrix} w_{ref} \\ u_{dref} \end{bmatrix} = \begin{bmatrix} \frac{1}{\tau_p} & 0 \\ 0 & \frac{1}{\tau_q} \end{bmatrix} \begin{bmatrix} \begin{bmatrix} P_{ref} \\ Q_{ref} \end{bmatrix} - \begin{bmatrix} P_{mes} \\ Q_{mes} \end{bmatrix} \end{bmatrix} - \begin{bmatrix} D\Delta w \\ 0 \end{bmatrix}$$

$$+ \begin{bmatrix} w_0 \\ U_0 \end{bmatrix} \tag{10}$$

To better fuse MFRL with the existing microgrid control framework, it is necessary to first know the capabilities of each MFRL algorithm, and then choose the proper algorithms in real applications. Thus, the following sections introduce the map of MFRL, the features of main stream MFRL algorithms, and how MFRL can be incorporated into the existing microgrid control framework.

III. Model-Free Reinforcement Learning

This section first gives a brief introduction to RL and then summarizes the methodology of MFRL.

A. Formulation of RL

RL is a basic ML paradigm formulated as an MDP. As shown in Fig. 3(a), the environment defines the state space $S$ and the agent holds the action space $A$. The agent keeps interacting with the environment to update its policy $\pi$ that maps the environment states to actions. In each iteration, the agent chooses action $a_t \in A$ according to $\pi$. Then, the environment generates the next state according to its intrinsic transition probability $P(s_{t+1} \mid s_t, a_t) : S \times A \rightarrow \Delta(S)$ and feeds back

C. Motivation for MFRL

1) Challenges in the Existing Control Framework: The high-level research map and modularized control blocks clearly show how existing microgrids are controlled. However, the evolution of microgrids brings more challenges to the existing control framework. The challenges are five-fold: i). The penetration of DERs results in higher uncertainties. Although some robust and stochastic techniques have been employed to address the emerging uncertainties, they are somehow conservative and the probability distribution function still needs to be accurately estimated. ii). It is difficult to model each element of microgrids in detail, i.e., customer behavior. The information that is difficult to model is critical for energy management in M5. iii). Some system parameters are not always accessible; even if accessible, they are not necessarily accurate. iv). Microgrid dynamics are becoming faster because more and more inverter-based resources participate in grid services by adaptively changing their control modes and control parameters. Then, the existing controllers may not be valid anymore. v). Smart grids call for autonomous microgrids, with which engineers and grid operators are free from parameter tuning for modules in Fig. 2. Even for other model-free controllers, they still need elaborate tuning for hyper-parameters, i.e., the membership functions of the fuzzy logic controller and the coefficients of the adaption law.

2) Why MFRL?: Microgrid operators have access to massive data sampled by phasor measurement units (PMUs) and advanced metering infrastructures (AMIs) now [63]. It opens the possibility for data-driven control. MFRL is an advanced decision-making technique with goal-oriented, data-driven, and model-free characteristics [64]. With the help of MFRL, the uncertainties of the model and parameters may be mitigated through repeated interaction between the environment and the RL agent. It is also beneficial to the autonomous operation of microgrids because the RL agent can actively update its policy based on the microgrid dynamics.

Readers are encouraged to check Refs. [56], [57] for some modified VSG and droop control techniques that provide more effective inertia support to microgrids.

5) M5: Auxiliary Service ∪ Optimization Module: Microgrids exploiting M1-M4 can withstand normal disturbances such as load changes and plug-and-play generations. Then, M5 participates in grid optimization and provides auxiliary services, i.e., optimized active and reactive power sharing, demand-side management, and V-f support [58]. In order for more economic energy management, M5 also calculates the steady-state setting points such as $(P_0, Q_0)$ by solving optimal power flow [59]. On the other hand, it generates the supplementary signals for controller parameters and outputs [60] according to the targets of auxiliary service. Review papers regarding M5 can be seen in [61], [62].
Fig. 3. The framework and map of MFRL. (a) agent-environment interaction in an MDP. (b) methodology.

the instant reward \(r(s_t, a_t)\) to the agent. The iteration is repeated until the agent finds the optimal policy \(\pi^*\) as follows.

\[
\pi^* \in \arg \max_{\pi} J(\pi) = \mathbb{E}_{\pi} \sum_{t=1}^{\infty} \gamma^t r(s_t, a_t)
\]

where \(\gamma\) is the discounting factor and \(J(\pi)\) is the infinite horizon discounted reward. The optimal policy guarantees the maximum accumulated reward obtained from the environment.

In MFRL, \(A\) and \(S\) can be either continuous or discrete. For the sake of illustration, this paper uses discrete notation to introduce the methodology.

B. Methodology of MFRL

Fig. 3(b) shows the mainstream MFRL methodology. They are categorized into value-based and policy-based algorithms.

1) Value-Based Algorithms: The value-based methods learn the Q-function that estimates the Q-value of state-action pairs \((s, a) \in S \times A\). The Q-function is denoted as \(Q_\pi\), based on which the agent can choose the optimal actions with the maximum Q-value. According to the Bellman equation,

\[
Q_\pi(s_{t+1}, a_{t+1}) = r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1}, a_{t+1}}[Q_\pi(s_{t+1}, a_{t+1})]
\]

Through temporal-difference learning, \(Q_\pi\) can finally converge to its true value under mild assumptions [65].

\[
Q_\pi(s_t, a_t) = Q_\pi(s_t, a_t) + \alpha \left[ r_t + \gamma \max_{a_{t+1} \in A} Q_\pi(s_{t+1}, a_{t+1}) - Q_\pi(s_t, a_t) \right]
\]

The approximated \(Q_\pi\) was first recorded in a Q-table [66]. Considering the table’s inefficiency, the deep Q-learning network (DQN) [67] replaced the Q-table with a deep artificial neural network (ANN), which has a strong fitting capability that maps the states to Q-value with less memory. Then, the DQN was further improved using the following tricks [68]:

- (Prioritized) Reply Buffer enhances the training efficiency.
- Double Network relieves the overestimation of Q-value.
- Dueling Network improves the performance in high-dimensional action space.

Later, a distributional DQN [69] and a quantile regression DQN [70] were proposed using stochastic policy and distributed training, and they were combined as ‘Rainbow DQN’ by David Silver [71] in 2017.

2) Policy-Based Algorithms: Policy gradient methods directly learn the parameterized policy based on feedback from the environment. Before diving into policy gradient algorithms, it is necessary to introduce the actor-critic (AC) structure. The AC structure has two ANN models that optionally share parameters: i) Critic updates the parameters of value functions; ii) Actor updates the policy parameters under the guidance of the critic. Under the AC structure, policy function can be either stochastic or deterministic. The stochastic policy is modeled as a probability distribution: \(a \sim \pi_\theta(a \mid s)\), while the deterministic policy is modeled as a deterministic decision: \(a = \pi_\theta(s)\). They classify the policy-gradient methods.

a) Stochastic Policy: As for stochastic policy \(a \sim \pi_\theta(a \mid s)\), the gradient of the expected reward to policy parameters is calculated according to policy gradient theorem [72] as follows

\[
\nabla J(\theta) = \sum_{s \in S} \mu_\theta(s) \sum_{a \in A} \pi_\theta(a \mid s)Q_\pi(s, a)\nabla \ln \pi_\theta(a \mid s)
\]

where \(\mu_\theta(S) \in \Delta(S)\) is the state distribution. Then, the policy is updated using the gradient ascent method

\[
\theta_{t+1} = \theta_t + \eta \nabla J(\theta)
\]

where \(\eta\) is the learning rate. It is necessary to avoid large updating of step size in each iteration since the policy gradient readily falls into a local maximum. To make the policy gradient training more stable, trust region policy optimization (TRPO) added a Kullback–Leibler (KL) divergence constraint to the process of policy updating [73]. It solves the optimization problem as follows

\[
\max_{\theta} \quad J(\theta) = \mathbb{E} \left[ \frac{\pi'_\theta(a \mid s)}{\pi_\theta(a \mid s)} A_0(a \mid s) \right] \\
\text{s.t.} \quad \mathbb{E} \left[ D_{KL}(\pi'_\theta \| \pi_\theta) \right] \leq \delta
\]

where \(\pi'_\theta\) is the new policy; \(D_{KL}\) is the KL-divergence.

Considering the complexity of measuring \(D_{KL}\) in each update, proximal policy optimization (PPO) was developed to accelerate the training [74]. PPO uses a clipped surrogate objective while retaining similar performance as follows

\[
\nabla J(\theta) \approx \frac{1}{\sum_{s \in S} \mu_\theta(s)} \sum_{s \in S} \mu_\theta(s) \sum_{a \in A} \pi_\theta(a \mid s)Q_\pi(s, a)\nabla \ln \pi_\theta(a \mid s)
\]

where \(\eta\) is the learning rate. It is necessary to avoid large updating of step size in each iteration since the policy gradient readily falls into a local maximum. To make the policy gradient training more stable, trust region policy optimization (TRPO) added a Kullback–Leibler (KL) divergence constraint to the process of policy updating [73]. It solves the optimization problem as follows

\[
\max_{\theta} \quad J(\theta) = \mathbb{E} \left[ \frac{\pi'_\theta(a \mid s)}{\pi_\theta(a \mid s)} A_0(a \mid s) \right] \\
\text{s.t.} \quad \mathbb{E} \left[ D_{KL}(\pi'_\theta \| \pi_\theta) \right] \leq \delta
\]

where \(\pi'_\theta\) is the new policy; \(D_{KL}\) is the KL-divergence.
max \( J(\theta) = \mathbb{E} \left[ \min \left\{ \pi_{\theta}'(a | s), \tilde{A}_{\theta}(a | s) \right\} \right] \) \\
\text{clip} \left\{ \frac{\pi_{\theta}'(a | s)}{\pi_{\theta}(a | s)}, 1 - \varepsilon, 1 + \varepsilon \right\} \tilde{A}_{\theta}(a | s) \right\} \right\} \}

(17)

In PPO, the actor network and critic network share the same learned features, and this may result in conflicts between competing objectives and simultaneous training. Hence, a phasic policy gradient (PPG) separates the training phased for actor and critic networks [75], which leads to a significant improvement in sampling efficiency. Other improved versions of the AC structure include advantage actor-critic (A2C), asynchronous advantage actor-critic (A3C), and soft actor-critic (SAC). A2C and A3C both enable parallel training using multiple actors, but the actors of A2C work synchronously, and those of A3C work asynchronously [76]. SAC improves the exploration of agents incorporating policy entropy [77].

b) Deterministic Policy: The gradient of deterministic policy \( a = \pi_{\theta}(s) \) is expressed as

\[
\nabla J(\theta) = \mathbb{E}_{s \sim \mu_{\theta}} \nabla_{a} Q_{\pi_{\theta}}(s, a) \left|_{a = \pi_{\theta}(s)} \right. \nabla_{a} Q_{\pi_{\theta}}(s)
\]

(18)

The deterministic policy gradient (DPG) method firstly used deterministic policy [78]. Then, the deep deterministic policy gradient (DDPG) was developed by combining the DPG and DQN [79]. The DDPG extends the discrete action space of the DQN to continuous space while learning a deterministic policy. Later, the twin delayed deep deterministic (TD3) policy gradient applied three tricks, i.e., clipped network, delayed update of critic network, and target policy smoothing to prevent the overestimation of \( Q \)-value in the DDPG.

3) Summary: The DQN, DDPG, and A3C are three basic paradigms of MFRL representing value-based methods, deterministic policy methods, and stochastic policy methods. Their upgraded versions, the Rainbow DQN, TD3, and PPG, SAC represent the state-of-the-art of each paradigm, which are the best choices for fusing MFRL with the existing microgrid control framework. Moreover, the value-based methods such as DQN are more suitable for discrete control tasks like transformer tap and switch on/off control, while the policy-based methods like TD3 are more suitable for continuous tasks such as active power and reactive power reference generation.

IV. FUSION OF MODEL-FREE REINFORCEMENT LEARNING WITH MICROGRID CONTROL

Section II and Section III introduce the existing microgrid control framework and the MFRL, separately. This section further the fusion details, including the application guidelines and the challenges and insights of using MFRL in microgrid control.

A. Application Guideline

1) Problem Formulation: Microgrid control is intrinsic to an infinite MDP that MFRL can solve. Ref. [80] answered the question of ‘How’, that is, ‘How to formulate a control problem that can be solved by MFRL?’ which includes four steps: i) Determine the environment, state space \( S \), and action space \( A \); ii) Design reward function \( R \) according to control targets; iii) Select proper learning algorithm; iv) Train agent and validate the learned policy. The four steps are exemplified below based on two specific application scenarios, frequency regulation and voltage regulation.

i) Formulation of frequency regulation: Eqs. (19)-(21) show the general configuration of a MFRL agent for frequency regulation in microgrids. The agent has unique action space when fusing with different modules in Fig. 2.

\[
S_f := \left\{ (w_i)_{i \in N}, (P_{ij}, Q_{ij})_{j \in I} \right\} \quad (19)
\]

\[
A_f = \begin{cases} 
M2 : \left[ (e_{abc,i})_{i \in I} \right] \\
M3 : \left[ (i_{d,i}, i_{q,i})_{i \in I} \right] \\
M4 : \left[ (w_{ref,i}, Q_{ref,i})_{i \in I_{GFL}} \right] \\
M5 : \left[ (P_{0,i}, Q_{0,i}, \Delta x_i)_{i \in I_{GFM}} \right]
\end{cases}
\]

(20)

\[
R_f(t) = - \sum_{i \in N} [w_i(t) - w_0]^2
\]

(21)

where \( w_i \) is frequency at each bus \( i \); \( P_{ij}, Q_{ij} \) is the power flow over line from bus \( i \) to bus \( j \); M2-M5 are the modules summarized in Fig. 2; \( I \) is the inverter set; \( I_{GFL} \) and \( I_{GFM} \) are the set of GFL inverters and GFM inverters, respectively. Since the control target is to maintain frequency, the deviation of frequency is designed as reward function.

ii) Formulation of voltage regulation: Eqs. (22)-(24) show the general configuration of a MFRL agent for frequency regulation in microgrids.

\[
S_v := \left\{ (v_i)_{i \in N}, (P_{ij}, Q_{ij})_{j \in I} ; (\tau_i)_{i \in T} \right\}
\]

(22)

\[
A_v = \begin{cases} 
M2 : \left[ (e_{abc,i})_{i \in I} \right] \\
M3 : \left[ (i_{d,i}, i_{q,i})_{i \in I} \right] \\
M4 : \left[ (w_{ref,i}, Q_{ref,i})_{i \in I_{GFL}} \right] \\
M5 : \left[ (P_{0,i}, Q_{0,i}, \Delta x_i)_{i \in I_{GFM}} \right]
\end{cases}
\]

(23)

\[
R_v(t) = - \sum_{i \in N} [v_i(t) - v_0]^2
\]

(24)

where \( v_i \) is the voltage magnitude of bus \( i \), and \( \tau_i \) is the tap positions of the on-load tap changers (OLTPs) of transformers. Compared with frequency regulation, the agent has distinct action of OLTPs in M5 for voltage regulation.

After selecting \( S, A, \) and \( R \), the mainstream MFRL algorithms are selected to update the policy of the agent. Note that the selected algorithms should be applicable to the application scenarios. For example, the discrete algorithm in Fig. 3(b) is better for discrete control actions like OLTPs. In addition, the above formulations give a general form of configuring an MFRL agent for microgrid control, and they can be modified according to customized control tasks.

In addition to problem formulation, there are another two fundamental questions regarding ‘What’ that remain to be answered. They are

* Q1: What kinds of tasks is MFRL suitable for?
* Q2: How can MFRL be fused with the existing microgrid control framework?
The following two subsections tries to answer these two questions based on the state-of-the-art of MFRL. The answers can serve as the application guideline for adopting MFRL in microgrids.

2) What Kinds of Tasks Is MFRL Suitable For?: In general, MFRL is suitable for tasks with the following four features:

i) Relatively unchanged environment. Policy learned by RL agents reflects the physical law in the training environments, which fundamentally determines the state transition probability. As shown in the diagram in Fig. 3(a), environment generates rewards based on \( P(s_{t+1} \mid s_t, a_t) : \mathcal{S} \times \mathcal{A} \rightarrow \Delta(\mathcal{S}) \) and feed the rewards to RL agent for policy updating. A new environment has distinct state transition probability function, which may have conflicts with the buffer data and trained policy. Thus, the working environment should not differ too much from the training environment. That’s why in Tab. I, the training microgrids and validation microgrids usually have fixed topology and predefined disturbances.

ii) Clear control target. Clear control targets facilitate the design of reward functions. The objective function in the optimization problem, optimal control, and MPC can be directly transformed to a reward function. With the function grouping and hierarchical structure in Fig. 1, the specific control targets can be briefly categorized into frequency regulation, voltage regulation equation, and economic benefits. Then, the voltage deviation [81], frequency deviation [87] and energy management cost/revenue [83], [84] are transformed into reward functions in (21) and (24). Crucially, a well-designed reward function gives the MFRL agent the best guide to learn the optimal policy.

iii) Available data. Environmental data must be accessible if the agent interacts with a real system. Also, the real environment should tolerate improper actions for exploration. If the environment is a simulator, the simulation should run quickly to allow for thousands of repetitions. For example, a fast a simulator and a real tokamak vessel were developed for training and validation in [30].

iv) Acceptable control complexity. ‘Acceptable’ means that the control complexity should be neither too low nor too high. For each perspective summarized in the high-level research map, there is no research trying to replace all the controllers. Most of the research just focused on a specific task that a model-based controller cannot handle but MFRL can, because there is no need to replace a simple model-based controller that has good performance and it is unrealistic to let AI directly control the whole microgrid for now.

3) How Can MFRL Be Fused With the Existing Microgrid Control Framework?: MFRL is essentially a useful tool that serves microgrid control. It follows microgrid control targets when fused with the existing control framework. In general, there are three ways of fusing as follows.

i) Model identification and parameter tuning. MFRL assists in identifying the uncertain models of the grid components accurately. Also, it can address the uncertainty and unavailability of model parameters and release the grid operators from complex and time-consuming parameter tuning, especially tuning a large model with many parameters.

ii). Supplementary signal generation. MFRL can generate the supplementary control signals for model-based controllers, with which the current controllers can be made more robust and deal with complicated control tasks.

iii). Controller substitution. MFRL can completely replace the existing model-based controllers if they are no longer effective. It needs fewer inputs but has better performance than model-based controllers owing to the ANN’s strong fitting capability.

In general, the application guide is summarized based on the existing microgrid control research that employ MFRL. The detailed literature review will be performed in the next subsection.

B. Literature Review

Sorted in the way of fusing, Table I summarizes the literature adopting MFRL in microgrids, where the key features are listed in the last column. In general, MFRL has fused with the optimization and control tasks in microgrids. Most research has tried to replace the existing model-based controllers with MFRL agents. In addition, more researchers focus on optimization problems that have clear targets. The objective functions are directly transformed or incorporated into the reward function.

C. Challenges and Insights

Although many researchers have been investigating the applications of MFRL in microgrid control, there is still a clear gap between theory (simulation) and practice (real microgrid operation). The main concerns are the aspects of the environment, scalability, generalization, security, and stability. This subsection summarizes these challenges and gives some insights on how to tackle them.

1) Environment:

- Challenges: As shown in Fig. 4, the conventional model-based microgrid controllers have several types of tests before implementation, i.e., simulation, controller hardware in the loop (HIL) test, power HIL test, subscale system test, and full system test. They are the options for the MFRL environment.
Existing literature suggests offline training in the numerical simulator and online implementation in real systems [95] because the RL agent requires sufficient exploration during training which is unrealistic in HIL or real systems. That’s why early RL was mainly used in video games, where the simulator could perfectly emulate the working environments. Among the current power testbed types, simulation has the highest coverage of test scenarios but the

<table>
<thead>
<tr>
<th>Ref.</th>
<th>Year</th>
<th>Topic</th>
<th>Algorithm</th>
<th>Way of fusing</th>
<th>Environment</th>
<th>Key features</th>
</tr>
</thead>
<tbody>
<tr>
<td>[82]</td>
<td>2020</td>
<td>Converter voltage stability</td>
<td>PPO</td>
<td>Parameter tuning for M4</td>
<td>dSPACE</td>
<td>1) Adaptively tune the feedback gains of the ultra-local model; 2) Mitigate the stability issues caused by constant power loads</td>
</tr>
<tr>
<td>[86]</td>
<td>2022</td>
<td>Cyber attack</td>
<td>DQN, Multi-agent DDPG</td>
<td>Model identification for M2-M4</td>
<td>MATLAB/Simulink, dSPACE, MicroLabBox, DS1202</td>
<td>1) RL agent automatically discover the vulnerable spots and generate coordinated destabilizing false data attacks; 2) Enhanced agent with a sniffing feature to enable maintaining the stealthy attacks under connection failure.</td>
</tr>
<tr>
<td>[85]</td>
<td>2021</td>
<td>Transient stability</td>
<td>SAC</td>
<td>Model identification for M4</td>
<td>Numerical simulator</td>
<td>1) Consider multiple events that result in transient stability simultaneously; 2) Test learned policy in new events</td>
</tr>
<tr>
<td>[89]</td>
<td>2020</td>
<td>Microgrid Penetration Test</td>
<td>A3C</td>
<td>Supplementary signal generation for M5</td>
<td>Numerical simulator</td>
<td>1) Perform Penetration Testing for microgrids 2) RL agent uncovers the malicious input that can compromise the effectiveness of the controller</td>
</tr>
<tr>
<td>[88]</td>
<td>2021</td>
<td>dc-dc buck converter control</td>
<td>DDPG</td>
<td>Supplementary signal generation for M4</td>
<td>dSPACE, MicroLabBox and DS1302 I/O board</td>
<td>1) Design an intelligent PI controller based on sliding mode observer to mitigate instability; 2) RL agent generates the auxiliary signals to reduce the error of observer</td>
</tr>
<tr>
<td>[92]</td>
<td>2021</td>
<td>Secondary frequency control</td>
<td>DDPG</td>
<td>Supplementary signal generation for M5</td>
<td>Matlab and dSPACE 1202 board</td>
<td>1) Consider Type-II fuzzy system; 2) Generate supplementary signals for PI-based secondary controllers</td>
</tr>
<tr>
<td>[90]</td>
<td>2022</td>
<td>Secondary frequency control</td>
<td>A2C</td>
<td>Supplementary signal generation for M4</td>
<td>Discrete-time ODE model, numerical simulator</td>
<td>1) Achieve frequency synchronization within an ultimate bound given dominantly resistive and/or inductive line and load impedances; 2) A feedback control is formulated based on the unknown dynamics, using Lyapunov theory.</td>
</tr>
<tr>
<td>[91]</td>
<td>2021</td>
<td>Mode transition</td>
<td>Q-learning</td>
<td>Supplementary signal generation for M4</td>
<td>Numerical simulator</td>
<td>1) RL agent generates voltage angle and magnitude adjustments; 2) Enables bulk power grid restoration by using microgrids’ black start capability.</td>
</tr>
<tr>
<td>[94]</td>
<td>2019</td>
<td>Battery SOC control</td>
<td>DDPG</td>
<td>Controller substitution in M4</td>
<td>Numerical simulator</td>
<td>1) Perform supervised pre-training for critic-network based on control cost; 2) Perform pre-training for actor-network based on the output of PI controllers</td>
</tr>
<tr>
<td>[95]</td>
<td>2019</td>
<td>Energy storage system control</td>
<td>Q-learning</td>
<td>Controller substitution in M4 and M5</td>
<td>Matlab-Simulink Simscape toolbox</td>
<td>1) Optimize the charging and discharging profile to suppress the disturbance caused by integrating a new hybrid energy system; 2) One network estimates the unknown system dynamics and the other solves the optimal policy</td>
</tr>
<tr>
<td>[96]</td>
<td>2020</td>
<td>Emergency control</td>
<td>DQN</td>
<td>Controller substitution in M5</td>
<td>InterPSS in java and OpenAI in python</td>
<td>1) Train RL agent under the circumstance of predefined topology and random short-circuit faults; 2) Use hybrid simulation with java and python</td>
</tr>
<tr>
<td>[97]</td>
<td>2021</td>
<td>Islanding transition control</td>
<td>Q-learning</td>
<td>Controller substitution in M3-M5</td>
<td>Matlab-Simulink</td>
<td>1) Update specific values or parameters in reinforcement learning with artificial emotion; 2) Implement load shedding to reduce the impacts of unintentional islanding</td>
</tr>
<tr>
<td>[98]</td>
<td>2021</td>
<td>Energy storage system control</td>
<td>DQN</td>
<td>Controller substitution in M4</td>
<td>Numerical simulator (TensorFlow and GUROBI)</td>
<td>1) Improve robustness with prioritized replay policy based on sum-tree; 2) RL agent directly outputs actions without solving an optimization problem</td>
</tr>
<tr>
<td>[4]</td>
<td>2022</td>
<td>Peer-to-peer energy trading</td>
<td>Multi-agent TD3</td>
<td>Controller substitution in M5</td>
<td>Numerical simulator</td>
<td>1) Consider both external peer-to-peer energy trading and internal energy conversion; 2) The high-dimensional decision-making problem is solved by multi-agent TD3 under the resolution of hours</td>
</tr>
<tr>
<td>[81]</td>
<td>2021</td>
<td>Secondary voltage control</td>
<td>Multi-agent A2C</td>
<td>Controller substitution in M</td>
<td>Numerical simulator</td>
<td>1) A high-fidelity powergrid simulation platform ‘PGSim’ is developed as environment; 2) Enhance the scalability with spatial discount factor to reduce the effect from remote agents.</td>
</tr>
<tr>
<td>[83]</td>
<td>2022</td>
<td>P2P trading</td>
<td>Multi-agent DDPG</td>
<td>Controller substitution in M5</td>
<td>Numerical simulator</td>
<td>1) Incentivize energy trading with distribution network tariffs to satisfy the physical constraints; 2) Do simulations based on real-world trading data sets.</td>
</tr>
<tr>
<td>[99]</td>
<td>2022</td>
<td>Energy management</td>
<td>Trust region PPO</td>
<td>Controller substitution in M5</td>
<td>Numerical simulator</td>
<td>1) The distribution system operator (DSO) is viewed as a RL agent without knowing user information for privacy protection; 2) Integrate a differentiable trust region layer to improve the robustness of the policy updating.</td>
</tr>
<tr>
<td>[100]</td>
<td>2021</td>
<td>Online economic dispatch</td>
<td>Hierarchical Q-learning</td>
<td>Controller substitution in M5</td>
<td>Numerical simulator</td>
<td>1) Offers a subtle blend of immediate and future rewards to guarantee a long-term performance; 2) Integrate domain knowledge to narrow down learning space to a feasible region and avoids violations.</td>
</tr>
</tbody>
</table>
least fidelity, which is the major concern of employing MFRL. Even if the agent learned a good policy in a numerical simulator, it may not function effectively in a real microgrid.

- **Insights:** As for numerical simulators, they are on the way to developing a more accurate and faster toolbox capable of serving as a high-fidelity MFRL environment. Improved power system modeling [101], [102] and more efficient numerical simulation techniques, such as the hybrid symbolic-numeric framework [103], are currently being developed. Further, it would be better to develop a standardized and personalized training environment that assists in setting up the interface with power simulators such as PSCAD, PSSE, and MATLAB-Simulink, just like “Gym” in the field of deep RL. The standardized environment can also serve as a baseline for algorithm tests and comparisons. On the other hand, it is a good way to design a HIL test system that is equipped with specialized protection and can tolerate random exploration to some degree. In this way, the HIL test system may work as an environment that closely resembles an actual microgrid. Moreover, MFRL agent can learn from historical data. To improve the learning efficiency and address the problem of real-data insufficiency, some advanced techniques have been developed. For example, i) long-tail learning [104] can learn effectively on biased data set; ii) deep active learning [105] can also be used to more efficiently label disturbance data.

2) **Scalability:**

- **Challenges:** MFRL suffers from the curse of dimensionality like some model-based controllers. The expansion of state space and action space will result in an exponential increase in control complexity, thereby increasing the difficulty of exploration and training. Existing MFRL research on microgrid control mainly focuses on some small-scale problems [98] and utilizes ANN with a few layers. To promote the application of MFRL in microgrid control, it is necessary to improve its scalability.

- **Insights:** On the one hand, it is an effective way to reduce control complexity by integrating domain knowledge into problem formulation. For example, [106] narrowed down the learning space and avoided baseline violations based on the generation constraints. On the other hand, it would be better to increase the capability of existing MFRL models by: i) increasing the exploration efficiency by designing guided exploration strategies like evolutionary RL [107]; ii) increasing the fitting capability of ANN through the modern design of network structures, i.e., sequential-to-sequential networks and transformers [108]; iii) increasing the training efficiency through distributed techniques like federated learning [109] and edge computing [110]. All of these methods can help relieve the pressure on training and make MFRL more scalable for microgrid control.

3) **Generalization:**

- **Challenges:** Similar to DL, MFRL was accused of “inability of generalization” because a well-trained agent does not function effectively in a changing environment [111]. Even in an unchanged environment, the diversity of disturbances may also distort the agent. In microgrid control, it is difficult to cover all the disturbances during the training, which is critical on the condition that RL agents replace the existing controllers.

- **Insights:** Firstly, rich training scenarios benefit the generalization of MFRL. For example, [112] addressed the uncertainty of Volt-Var control in active distribution systems by generating a bunch of offline training scenarios. It is also a good way to employ robust RL that can tolerate the uncertainty of the environment [113]. Further, transfer learning can also enhance the MFRL’s generalization capability, which has proven to be effective in the field of DL.

4) **Security:**

- **Challenges:** Security is referred to as static security in this paper, meaning that system state should respect the static physical constraints to avoid damaging the device. In microgrids, these constraints can be thermal limit constraints and control signal constraints decided by the physical capability of microgrid components. They are usually explicit and known according to microgrid device manufacture, and there are IEEE Standards setting the secure operational range of voltage and frequency. However, due to the non-interpretability of ANN, the learned policy cannot always guarantee each variable respect the constraints. Furthermore, it is also a problem to guarantee secure exploration in a HIL or real system. In the future, MFRL agents may be trained in a HIL microgrid to overcome the shortcomings of numerical simulators, where the exploration cannot violate the physical constraints of the HIL or real system for sure.

- **Insights:** Through constrained RL [28], [114] and safe RL [115], [116], the actions of RL agents can be projected to a safety region and thus always respect the physical operational constraints. In addition, physics-constrained and physics-informed deep learning [117] is also under development and can be integrated into MFRL to address security concerns. In physics-constrained deep learning, a “safety layer” is often leveraged to maintain constraint satisfaction under different physics knowledge, while physics-informed learning embeds the knowledge of physical laws that govern by partial differential equations into training.

5) **Stability:**

- **Challenges:** Stability is referred to as dynamic stability under a disturbance. According to the definition in [118], the stability is the ability of an electric power system, for a given initial operating condition, to regain a state of operating equilibrium after being subjected to a physical disturbance, with most system variables bounded so that practically the entire system remains intact. Model-based microgrid controllers must pass the stability test through eigenvalue analysis or the Lyapunov function validation before implementation. However, the employment of MFRL challenges the model-based criteria because the uninterpretable RL agents dramatically change the closed-loop dynamics of microgrids.

- **Insights:** Integrating domain knowledge is the best way to guarantee microgrid stability for now. As for the first two fusing approaches, i) model identification and parameter tuning and ii) supplementary signal generation, model-based stability criteria can still be used to verify the system stability because the MFRL agent doesn’t break down closed-loop...
systems. MFRL complements the model-based approaches and improves them in a data-driven way. The supplementary signals generated by the MFRL agent can be viewed as hyper-parameters. Through techniques like semi-definite programming (SDP), linear matrix inequality (LMI), and sum-of-square programming [119], the security range of these hyper-parameters can be obtained to guarantee dynamic stability [120]. As for the third way of fusion, the complete controller substitution, MFRL agents dramatically change the closed-loop dynamics and make the system difficult to model. To address the stability issues in this condition, this paper gives three potential solutions. i) enrich the training data and training scenarios. The learned policies basically reflect the state transition of the environment. If the training data set has covered sufficient instability scenarios, the corresponding punishment reward can help RL agents avoid unstable actions. ii) use a physics-informed approach by integrating model-based stability criteria into MFRL training. For example, the Lyapunov function [121] and the Gaussian process estimation [116] can be used to generate stability criteria for MFRL training, and [122] proposed a Lyapunov-regularized RL for transmission system transient stability. iii) perform policy stability validation through time-domain simulation (TDS). TDS has been widely used in power systems to validate the stability of nonlinear components or modules. It can also help validate the stability of the inexplicable RL policy.

V. CONCLUSION

Model-based controllers are still the foundation of existing microgrid control systems. However, the emerging challenges caused by the uncertainty of DERs and extreme weather call for advanced control techniques. As a model-free and data-driven approach, MFRL opens the possibility of non-linear, high-dimensional, and high-complex microgrid control. It may contribute to a huge upgrade of the existing control framework. Against this background, this paper firstly performs a comprehensive review of the current microgrid control framework and then summarizes the applications of MFRL. In general, there are three ways of fusing MFRL with the existing model-based controllers, including i) model identification and parameter tuning, ii) supplementary signal generation, and iii) controller substitution. For now, there is still an obvious gap between the theory (simulation) and its practical application. The challenges are mainly categorized into environment, scalability, generalization, security, and stability. With the rapidly developed techniques in the fields of both power and artificial intelligence, the author believes the challenges summarized in this paper will finally be overcome. Someday in the future, the MFRL can perfectly fuse with the existing microgrid control framework.

REFERENCES


**Buxin She** (Student Member, IEEE) received the B.S.E.E. and M.S.E.E. degrees from Tianjin University, Tianjin, China, in 2017 and 2019, respectively. He is currently pursuing the Ph.D. degree with the Department of Electrical Engineering and Computer Science, The University of Tennessee at Knoxville, Knoxville. His research interests include microgrid operation and control, machine learning in power system, and power grid resilience.

**Fangxing Li** (Fellow, IEEE) is also known as Fran Li. He received the B.S.E.E. and M.S.E.E. degrees from Southeast University, Nanjing, China, in 1994 and 1997, respectively, and the Ph.D. degree from Virginia Tech, Blacksburg, VA, USA, in 2001. He is currently the James W. McConnell Professor of Electrical Engineering and the Campus Director of CURENT with the University of Tennessee at Knoxville, Knoxville, TN, USA. His current research interests include microgrid, and energy markets. He has received numerous awards and honors, including the R&D 100 Award in 2020, the IEEE PES Technical Committee Prize Paper Award in 2019, the five Best or Prize Paper Awards at international journals, and the six Best Papers/Posters at international conferences. He has been the Editor-in-Chief of *IEEE Open Access Journal of Power and Energy* since 2020. From 2020 to 2021, he served as the Chair of IEEE PES Power System Operation, Planning and Economics Committee. He has been serving as the Chair of IEEE WG on Machine Learning for Power Systems since 2019.

**Jingqiu Zhang** (Student Member, IEEE) received the B.S. and M.S. degree in electrical engineering from Tianjin University, Tianjin, China, in 2016 and 2019, respectively. He is currently pursuing the Ph.D. degree in electrical and computer engineering with the National University of Singapore, Singapore. His current research interests include cyber security of power grids, distributed control, and optimization in microgrids.

**Hantao Cui** (Senior Member, IEEE) received the B.S. and M.S. degrees in electrical engineering from Southeast University, China in 2011 and 2013, and the Ph.D. degree in electrical engineering from the University of Tennessee at Knoxville, Knoxville in 2018. He is currently an Assistant Professor with the School of Electrical and Computer Engineering, Oklahoma State University. His research interests include power system modeling, simulation, and high-performance computing.

**Rui Bo** (Senior Member, IEEE) received the B.S.E.E. and M.S.E.E. degrees in electrical power engineering from Southeast University, China, in 2000 and 2003, respectively, and the Ph.D. degree in electrical engineering from the University of Tennessee at Knoxville, Knoxville, in 2009. He is currently an Assistant Professor with the Electrical and Computer Engineering Department, Missouri University of Science and Technology (formerly University of Missouri–Rolla). He worked as a Principal Engineer and the Project Manager with Midcontinent Independent System Operator from 2009 to 2017. His research interests include computation, optimization and economics in power system operation and planning, high performance computing, electricity market simulation, evaluation, and design.