Fusion of Model-free Reinforcement Learning with Microgrid Control: Review and Insight

Buxin She, Student Member, IEEE, Fangxing Li, Fellow, IEEE, Hantao Cui, Senior Member, IEEE, Jingqiu Zhang, Student Member, IEEE, 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 six distinct perspectives, 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 insight.

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, a smaller system size [5], a greater degree of uncertainty [6], lower system inertia [7], 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 [8]. 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 [9]. Ref. [10] 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 [11], operation modes [12], and control techniques [13]. 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 [14]. Researchers who study microgrid control could benefit from this work. It could help them figure out the state-of-the-art in the field quickly and accurately.

Among the control techniques utilized in microgrid control, 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. 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. Due to its successful applications in video games [15], autonomous driving [16], robotics [17], and power systems [18], MFRL is gaining more and more attention. Recently, researchers from DeepMind and École Polytechnique Fédérale de Lausanne developed a non-linear, high-dimensional, and RL-based magnetic controller for nuclear fusion [19] and published their work in Nature. This indicates the great potential of implementing MFRL in 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, control level, 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 the high-level research map and modularized control blocks. Section III gives a brief introduction to RL and the mainstream algorithms of MFRL. 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) control level, 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 [20] - [21]. 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 measurements. Ref. [22] summarized the strategies for the seamless transition between GC and IS modes.

2) Control level: The 2nd viewpoint is associated with the control level. The terms GC and IS for microgrid operation modes change to GFL and GFM when it comes to inverters. Inverter-level control focuses on the control schemes of GFL and GFM inverters [23], while grid-level control focuses more on cooperation among clustered microgrids, i.e., accurate and economical power-sharing among networked microgrids [24].

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 sets up the control targets for all the controllers clearly, with which each level controller can work independently within the distinct timescales [8].

The zero controller deals with the modeling and control of inverters on the timescale of milliseconds. The indirect current control is used in the early stages [24], and is later replaced by the direct current control due to its fast response and accurate current control capability [26]. More details can be found in the review paper [27]. The primary controller is responsible for automatic power sharing among generations while maintaining V-f stability in the timescale of seconds. Because the primary controller pertains to the fast control actions, it predominantly determined the stability of microgrids [2]. Ref. [28] 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. [29] - [30] 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 [31] - [32].

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 [33].

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 adjacent nodes. Average-based, consensus-based, and event-triggered distributed algorithms are employed in microgrid control [34]. Distributed control algorithms require the connected communication graph of microgrids. They also have a reduced convergence speed as the network grows [35]. 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. [36] 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. [37] summarized the advances and opportunities of employing MPC in microgrids, and [38] 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. [39] conducted a survey on DL for microgrid load and DER foresting. A review of MFRL for microgrid control has yet to be done, which is why it is the main scope of this manuscript.

B. Configuration of grid-following and grid-forming inverters

GFL and GFM inverters are no doubt one of the most important units in microgrids [40]. This section 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.

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 point. 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 $e_{abc-ref}$ 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.

$$L_f \left[ \frac{d i_d}{dt} \right] + wL_f \left[ -i_q \right] = \left[ e_d \right] - \left[ u_d \right]$$

$$\left[ \begin{array}{c} e_d \\ e_q \end{array} \right] = \left[ \begin{array}{c} u_d \\ u_q \end{array} \right] + \left[ \begin{array}{c} -i_q \\ i_d \end{array} \right] \left( \begin{array}{c} k_{pid} + \frac{k_{pd}}{s} \\ k_{pq} + \frac{k_{iq}}{s} \end{array} \right) \left[ \begin{array}{c} i_{dref} \\ i_{qref} \end{array} \right]$$

3) M3: Current-ref module: The 3rd module (M3) is named the ‘Current-ref Module’ since it generates the reference current $[i_{dref}, i_{qref}]$ for M2. For a GFL inverter, $[i_{dref}, i_{qref}]$ are regulated based on the error between the actual output and the reference value. Eq. (3) depicts the transfer function of M3 using PI controllers.

$$\left[ \begin{array}{c} i_{dref} \\ i_{qref} \end{array} \right] = \left[ \begin{array}{c} k_{pI} + \frac{k_{pc}}{s} \\ k_{qI} \end{array} \right] \left[ \begin{array}{c} P_{mes} - P_{ref} \\ Q_{mes} - Q_{ref} \end{array} \right]$$

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 (4) and (5), respectively.

$$C_f \left[ \frac{du_d}{dt} \right] + wC_f \left[ -u_q \right] = \left[ i_d \right] - \left[ i_{gq} \right]$$

$$\left[ \begin{array}{c} i_{dref} \\ i_{qref} \end{array} \right] = \left[ \begin{array}{c} i_{gd} \\ i_{gq} \end{array} \right] + wC_f \left[ -u_q \right] + \left[ k_{pud} + \frac{k_{pu}}{s} 0 \right] \left[ \begin{array}{c} u_{dref} \\ u_{qref} \end{array} \right] - \left[ \begin{array}{c} u_d \\ u_q \end{array} \right]$$

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 [27]. 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 [41]. 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 [42].

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**

$$\left[ \begin{array}{c} P_{ref} \\ Q_{ref} \end{array} \right] = \left[ \begin{array}{c} k_P f_0 \\ k_v \end{array} \right] \left[ \begin{array}{c} w_0 \\ U_0 \end{array} \right] - \left[ \begin{array}{c} w_{mes} \\ U_{mes} \end{array} \right] + \left[ \begin{array}{c} P_0 \\ Q_0 \end{array} \right]$$

- **Droop-controlled GFM inverter**

$$\left[ \begin{array}{c} w_{ref} \\ u_{dref} \end{array} \right] = \left[ \begin{array}{c} 1 \\ k_f \end{array} \right] \left[ \begin{array}{c} 0 \\ 1 \end{array} \right] \left[ \begin{array}{c} P_0 \\ Q_0 \end{array} \right] - \left[ \begin{array}{c} P_{mes} \\ Q_{mes} \end{array} \right] + \left[ \begin{array}{c} w_0 \\ V_0 \end{array} \right]$$

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 [43]. 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**

$$\left[ \begin{array}{c} P_{ref} \\ Q_{ref} \end{array} \right] = \left[ \begin{array}{c} D + ks \\ 0 \end{array} \right] \left[ \begin{array}{c} w_0 \\ U_0 \end{array} \right] - \left[ \begin{array}{c} w_{mes} \\ U_{mes} \end{array} \right] + \left[ \begin{array}{c} P_0 \\ Q_0 \end{array} \right]$$

- **VSG-controlled GFM inverter**

$$\left[ \begin{array}{c} w_{ref} \\ u_{dref} \end{array} \right] = \left[ \begin{array}{c} \frac{1}{J_{ps}} \\ 0 \end{array} \right] \left[ \begin{array}{c} \frac{1}{2q_{qs}} \end{array} \right] \left[ \begin{array}{c} w_0 \end{array} \right] \left[ \begin{array}{c} P_{ref} \\ Q_{ref} \end{array} \right] - \left[ \begin{array}{c} P_{mes} \\ Q_{mes} \end{array} \right] - \left[ \begin{array}{c} \Delta w \\ 0 \end{array} \right] + \left[ \begin{array}{c} w_0 \\ U_0 \end{array} \right]$$
C. Motivation for MFRL

The modularized control blocks are constituted of model-based controllers. However, the emerging DERs and demand-side variation bring more uncertainties and challenges to the existing control framework.

On the one hand, it is difficult to model each element of microgrids in detail, i.e., customer behavior and regional weather. Some system parameters are not always accessible; even if accessible, they are not necessarily accurate. The tuning efforts also increase rapidly as the control complexity grows.

On the other hand, microgrid operators have access to massive data sampled by phasor measurement units (PMUs), advanced metering infrastructures (AMIs), and wide-area monitoring systems (WAMS) now [51]. It opens the possibility for data-driven control. MFRL is an advanced decision-making technique with goal-oriented, data-driven, and model-free characteristics. 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. Thus, the following sections introduce the map of MFRL and illustrates how MFRL can be incorporated into the bottom-level modularized control blocks.

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, the environment defines the state space $\mathcal{S}$ and the agent holds the action space $\mathcal{A}$. 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 $a_t \in \mathcal{A}$ according to $\pi$. Then, the environment generates the next state according to its intrinsic transition probability $P(s_{t+1} | s_t, a_t) : \mathcal{S} \times \mathcal{A} \rightarrow \Delta(\mathcal{S})$ and feeds back 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.

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

(10)
Rainbow DQN (2017) and a quantile regression DQN [57] were proposed using stochastic policy and distributed training, and they were combined as a ‘Rainbow DQN’ by David Silver [58] 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: \( a \sim \pi(\theta | s) \), while the deterministic policy is modeled as a deterministic decision: \( a = \pi_D(s) \). They classify the policy-gradient methods.

a) Stochastic Policy: As for stochastic policy \( a \sim \pi(\theta | s) \), the gradient of the expected reward to policy parameters is calculated according to policy gradient theorem [59] as follows

\[
\nabla J(\theta) = \sum_{s \in S} \sum_{a \in A} \pi_D(s) Q(\pi(s), a) \nabla \ln \pi_D(s) \nabla \theta \ln \pi_D(s) \nabla \theta
\]

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

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

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 [60]. It solves the optimization problem as follows

\[
\max_{\theta} J(\theta) = \mathbb{E} \left[ \frac{\pi'_\theta(a | s)}{\pi_\theta(a | s)} \delta \theta(a | s) \right]
\]

s.t. \( \mathbb{E} [D_{KL}(\pi'_\theta || \pi_\theta)] \leq \delta \)

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 [61]. PPO uses a clipped surrogate objective while retaining similar performance as follows

\[
\max_{\theta} J(\theta) = \mathbb{E} \left\{ \min \left[ \frac{\pi'_\theta(a | s)}{\pi_\theta(a | s)} \delta \theta(a | s), 1 - \varepsilon, 1 + \varepsilon \right] \delta \theta(a | s) \right\}
\]

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 [62], which leads to a significant improvement in sampling efficiency. Other improved versions of the AC structure include advantage actor-critic (A2C),

- Double Network relieves the overestimation of Q-value.
- Dueling Network improves the performance in high-dimensional action space.

Later, a distributional DQN [56] and a quantile regression DQN [57] were proposed using stochastic policy and distributed training, and they were combined as a ‘Rainbow DQN’ by David Silver [58] in 2017.
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 [63]. SAC improves the exploration of agents incorporating policy entropy [64].

b) Deterministic Policy: The gradient of deterministic policy 
\[ a = \pi_\theta(s) \] is expressed as
\[ \nabla J(\theta) = \mathbb{E}_{s \sim \mu_\theta} \nabla_a Q_\pi_\theta(s,a) \big|_{a=\pi_\theta(s)} \nabla_\theta \pi_\theta(s) \] (17)

The deterministic policy gradient (DPG) method firstly used deterministic policy [65]. Then, the deep deterministic policy gradient (DDPG) was developed by combining the DPG and DQN [66]. 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 [67].

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.

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

This section presents the application guideline of MFRL. The challenges and insights of using MFRL are also discussed.

A. Application guideline

Microgrid control is intrinsic to an infinite MDP that MFRL can solve. Ref. [68] answered the question of ‘How’, that is, ‘How to formulate a microgrid control problem that can be solved by MFRL?’, which includes four main steps: i) Determine the environment, state, and action; ii) Design reward function according to control targets; iii) Select proper learning algorithm; iv) Train agent and validate the learned policy. However, there are another two 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?

This subsection 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.

1) 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. Thus, the working environment should not differ too much from the training environment. Here, the environment is distinguished from the disturbance. A disturbance can change system states but cannot change the physical law or the state transition probability.

ii) Clear control target. A clear control target facilitates the design of reward function. The objection function in the optimization problem, optimal control, and MPC can be directly transformed into a reward function. 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.

iv) Acceptable control complexity. ‘Acceptable’ refers to the control complexity, which should be neither too low nor too high. On the one hand, MFRL is not practical in some ‘easy’ tasks where model-based methods have excellent performance. On the other hand, MFRL fails to learn the optimal policy if the control task is too complicated. Hence, it is necessary to identify the agent’s learning capability and make sure the control complexity is acceptable before employing MFRL.

2) 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 control 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. Due to the ANN’s strong fitting capability, MFRL needs fewer inputs but has better performance than model-based controllers.

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, and security. 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 [75] 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 [75]. Among the current power testbed types, simulation has the highest coverage of test scenarios but the 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 [80] and more efficient numerical simulation techniques, such as the hybrid symbolic-numeric framework [81], are currently being developed. Further, it would be better to develop a standardized and customized training environment that assists in setting up the interface with power simulators such as PSCAD, PSSE, and MATLAB-

| Ref. | Year | Topic                         | Algorithm | Way of fusing | Environment               | Key features                                                                 |
|------|------|-------------------------------|-----------|---------------|---------------------------|-----------------------------------------------------------------------------|
| 67   | 2021 | Transient stability           | SAC       | Model identification | Numerical simulator     | 1) Consider multiple events that result in transient stability simultaneously; 2) Test learned policy in new events |
| 68   | 2020 | Converter voltage stability   | PPO       | Parameter tuning | dSPACE MicroLabBox       | 1) Adaptively tune the feedback gains of the ultra-local model; 2) Mitigate the stability issues caused by constant power loads |
| 69   | 2020 | Microgrid Penetration Test    | A3C       | Supplementary signal generation | Numerical simulator     | 1) Perform Penetration Testing for microgrids 2) RL agent uncovers the malicious input that can compromise the effectiveness of the controller |
| 70   | 2021 | DC-DC buck converter control  | DDPG      | Supplementary signal generation | dSPACE MicroLabBox and DS1302 I/O board | 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 |
| 71   | 2021 | Secondary frequency control   | DDPG      | Supplementary signal generation | Matlab and dSPACE 1202 board | 1) Consider Type-II fuzzy system; 2) Generate supplementary signals for PI-based secondary controllers |
| 72   | 2019 | Energy management             | Q-learning| Controller substitution | Matlab and Python       | 1) Perform privacy-preserved response learning for multicrogids; 2) Implement Monte Carlo method for decision making |
| 73   | 2019 | Battery control SOC           | DDPG      | Controller substitution | Numerical simulator     | 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 |
| 74   | 2019 | Energy storage system control| Q-learning| Controller substitution | Matlab-Simulink Simscape toolbox | 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 |
| 75   | 2020 | Emergency control             | DQN       | Controller substitution | InterPSS in java and OpenAI in python | 1) Train RL agent under the circumstance of predefined topology and random short-circuit faults; 2) Use hybrid simulation with java and python |
| 76   | 2021 | Islanding transition control  | Q-learning| Controller substitution | Matlab-Simulink          | 1) Update specific values or parameters in reinforcement learning with artificial emotion; 2) Implement load shedding to reduce the impacts of intentional islanding |
| 77   | 2021 | Energy storage system control| DDQN      | Controller substitution | Numerical simulator (TensorFlow and GUROBI) | 1) Improve robustness with prioritized replay policy based on sum-tree; 2) RL agent directly outputs actions without solving an optimization problem |
| 78   | 2022 | Peer-to-peer energy trading   | Multi-agent TD3 | Controller substitution | Numerical simulator     | 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 |

Fig. 4: Microgrid testbeds [79] and MFRL environment
Simulink, just like “Gym” in the field of deep RL [82]. 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.

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 [78] 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, [83] 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 [84]; ii). increasing the fitting capability of ANN through the modern design of network structures, i.e., sequential-to-sequential networks and transformers [85]; iii). increasing the training efficiency through distributed techniques like federated learning [86] and edge computing [87]. 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 [88]. 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, [89] 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 [90]. Further, transfer learning can also enhance the MFRL’s generalization capability, which has proven to be effective in the field of DL [91].

4) Security: • Challenges: Security is the final important challenge. Model-based microgrid controllers must pass the security test through eigenvalue analysis or the Lyapunov function validation before implementation. However, due to the non-interpretability of ANN, the learned policy cannot always guarantee safe actions. 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: Integrating domain knowledge is the best way to increase the security of MFRL for now. Through constrained RL and safe RL [92], the actions of RL agents can always respect the physical operational constraints and stability criteria such as the Lyapunov function [93] and the Gaussian process estimation [94]. For example, [95] proposed Lyapunov-regularized reinforcement learning for power system transient stability. In addition, physics-constrained and physics-informed deep learning [96] is also under development and can be integrated into MFRL to address security concerns.

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. 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 fuse with the existing microgrid control framework.

REFERENCES

[1] D. E. Olivares, A. Mehrizi-Sani, A. H. Etemadi, C. A. Cañizares, R. Iravani, M. Kazemipour, A. H. Hajimiriagh, O. Gomis-Bellmunt, M. Saeedifard, R. Palma-Behnke et al., “Trends in microgrid control,” IEEE Transactions on Smart Grid, vol. 5, no. 4, pp. 1905–1919, 2014.
[2] M. Farrokhabadi, C. A. Canizares, J. W. Simpson-Porco, E. Nasr, L. Fan, P. A. Mendoza-Araya, R. Tonkoski, U. Tamrakar, N. Hatzigryiou, D. Lagos et al., “Microgrid stability definitions, analysis, and examples,” IEEE Transactions on Power Systems, vol. 35, no. 1, pp. 13–29, 2019.
[3] J. Liu, Y. Miura, H. Bevra, and T. Ise, “Enhanced virtual synchronous generator control for parallel inverters in microgrids,” IEEE Transactions on Smart Grid, vol. 8, no. 5, pp. 2268–2277, 2016.
[4] T. Chen, S. Bu, X. Liu, J. Kang, F. R. Yu, and Z. Han, “Peer-to-peer energy trading and energy conversion in interconnected multi-microgrid systems using multi-agent deep reinforcement learning,” IEEE Transactions on Smart Grid, vol. 13, no. 1, pp. 715–727, 2021.
[5] C. Ju, P. Wang, L. Goel, and Y. Xu, “A two-layer energy management system for microgrids with hybrid energy storage considering degrad- ation costs,” IEEE Transactions on Smart Grid, vol. 9, no. 6, pp. 6047–6057, 2017.
[6] M. Rezkallah, A. Chandra, B. Singh, and S. Singh, “Microgrid: configurations, control and applications,” IEEE Transactions on Smart Grid, vol. 10, no. 2, pp. 1290–1302, 2017.
[7] N. Nikmehr and S. N. Ravadanegh, “Optimal power dispatch of multi- microgrids at future smart distribution grids,” IEEE transactions on smart grid, vol. 6, no. 4, pp. 1648–1657, 2015.
[8] A. Bidram and A. Davoudi, “Hierarchical structure of microgrids control system,” IEEE Transactions on Smart Grid, vol. 3, no. 4, pp. 1963–1976, 2012.
[9] D. Wu, F. Tang, D. Tragicos, H. C. Vasquez, and J. M. Guerrero, “A control architecture to coordinate renewable energy sources and energy storage systems in islanded microgrids,” IEEE Transactions on Smart Grid, vol. 6, no. 3, pp. 1156–1166, 2014.
and impact assessment for hybrid renewable energy integration,” IEEE Access, 2021.

[50] Y. Gao and N. Yu, “Deep reinforcement learning in power distribution systems: Overview, challenges, and opportunities,” in 2021 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT). IEEE, 2021, pp. 1–5.

[51] X. Chen, G. Qiu, Y. Tang, S. Low, and N. Li, “Reinforcement learning for decision-making and control in power systems: Tutorial, review, and vision,” arXiv preprint arXiv:2102.01168, 2021.

[52] J. Tsitsiklis and B. Van Roy, “An analysis of temporal-difference learning with function approximation (technical report lids-p-2322),” Laboratory for Information and Decision Systems, 1996.

[53] J. N. Tsitsiklis, “Asynchronous stochastic approximation and q-learning,” Machine learning, vol. 16, no. 3, pp. 185–202, 1994.

[54] J. Chung, “Playing atari with deep reinforcement learning,” Comput. Ence, vol. 21, pp. 351–362, 2013.

[55] H. Van Hasselt, A. Guez, and D. Silver, “Deep reinforcement learning with double q-learning,” in Proceedings of the AAAI conference on artificial intelligence, vol. 30, no. 1, 2016.

[56] M. G. Bellemare, W. Dabney, and R. Munos, “A distributional perspective on reinforcement learning,” in International Conference on Machine Learning. PMLR, 2017, pp. 449–458.

[57] W. Dabney, M. Rowland, M. Bellemare, and R. Munos, “Distributional reinforcement learning with quantile regression,” in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32, no. 1, 2018.

[58] M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver, “Rainbow: Combining improvements in deep reinforcement learning,” in Thirty-second AAAI conference on artificial intelligence, 2018.

[59] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, “Policy gradient methods for reinforcement learning with function approximation,” Advances in neural information processing systems, vol. 12, 1999.

[60] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, “Trust region policy optimization,” in International Conference on Machine learning. PMLR, 2015, pp. 1889–1897.

[61] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” arXiv preprint arXiv:1707.06347, 2017.

[62] K. W. Cubbe, J. Hilton, O. Klimov, and J. Schulman, “Phasic policy gradient,” in International Conference on Machine Learning. PMLR, 2021, pp. 2020–2027.

[63] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, “Asynchronous methods for deep reinforcement learning,” in International conference on machine learning, PMLR, 2016, pp. 1928–1937.

[64] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, “Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor,” in International conference on machine learning. PMLR, 2018, pp. 1861–1870.

[65] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, “Deterministic policy gradient algorithms,” in International conference on machine learning. PMLR, 2014, pp. 387–395.

[66] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, “Continuous control with deep reinforcement learning:” arXiv preprint arXiv:1509.02971, 2015.

[67] E. O. Arwa and K. A. Folly, “Reinforcement learning techniques for optimal power control in grid-connected microgrids: A comprehensive review,” IEEE Access, vol. 8, pp. 208 992–209 007, 2020.

[68] S. Wang, R. Diao, C. Xu, D. Shi, and Z. Wang, “On multi-event co-calibration of dynamic model parameters using soft actor-critic,” IEEE Transactions on Power Systems, vol. 36, no. 1, pp. 521–524, 2020.

[69] M. Hajijossef, M. Andalibi, M. Gheisarnejad, H. Farsizadeh, and M.-H. Khooban, “Dc/dc power converter control-based deep machine learning techniques: Real-time implementation,” IEEE Transactions on Power Electronics, vol. 35, no. 10, pp. 9971–9977, 2020.

[70] C. Neal, H. Dagdougui, A. Lodi, and J. M. Fernandez, “Reinforcement learning based penetration testing of a microgrid control algorithm,” in Proceedings of the IEEE, vol. 109, no. 1, pp. 43–76, 2020.

[71] Y. Cui and B. Zhang, “Lyapunov-regularized reinforcement learning for power system transient stability,” IEEE Control Systems Letters, vol. 6, pp. 974–979, 2021.