Multi-Agent Reinforcement Learning with Safety Layer for Active Voltage Control

Yufeng Shi
University of Science and Technology of China
Hefei, China
shiyufeng@mail.ustc.edu.cn

Mingxiao Feng
University of Science and Technology of China
Hefei, China
fmxustc@mail.ustc.edu.cn

Minrui Wang
University of Science and Technology of China
Hefei, China
wangminrui0804@mail.ustc.edu.cn

Wengang Zhou
University of Science and Technology of China
Hefei, China
zhwg@ustc.edu.cn

Houqiang Li
University of Science and Technology of China
Hefei, China
lihq@ustc.edu.cn

ABSTRACT

The main goal of active voltage control is to keep the voltage of each bus in the grid within a safe range. With the increasing penetration of renewable and distributed energy sources in the grid, growing complexity, increasing uncertainty, and aggravating volatility bring great challenges to voltage control in modern power systems. Traditional algorithms can hardly guarantee real-time safe control to cope with these challenges. In recent years, substantial attention has been paid to the application of multi-agent reinforcement learning algorithms (MARL) to coordinate the control units in each area of the grid in real time for active voltage control in complex scenarios. However, these MARL algorithms do not explicitly guarantee that the power system satisfies the security constraints. There is a little in-depth study on safe multi-agent policy learning in multi-agent-based voltage control, especially the direct correction of unsafe actions. In this paper, we formalize the active voltage control problem as a Constrained Markov Game and approach it with a centralized data-driven safety layer that requires global observations and maps unsafe actions to safe actions. In order to make the policy network rely on local observations for decentralized execution, we introduce two novel components into the policy network: action correction penalty loss and action correction subnetworks. Notably, our approaches are easily extendable to other MARL algorithms for continuous actions. In the experiments, we evaluate our methods in the power distribution network simulation environment and demonstrate the capability of the safety layer to correct unsafe actions and the effectiveness of the safety layer to improve the performance of the policy itself.

KEYWORDS

Multi-Agent Deep Reinforcement Learning; Active Voltage Control; Safe Exploration

ACM Reference Format:
Yufeng Shi, Mingxiao Feng, Minrui Wang, Wengang Zhou, and Houqiang Li. 2023. Multi-Agent Reinforcement Learning with Safety Layer for Active Voltage Control. In *Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), London, United Kingdom, May 29 – June 2, 2023*, IFAAMAS, 9 pages.

1 INTRODUCTION

The utilization of renewable and distributed energy sources (solar energy, wind power, etc.) in the power grid is important to alleviate the current energy shortage and environmental protection problems [11], it also poses significant challenges to the safe operation of the grid. Control problems in power systems can be broadly classified into three categories: frequency regulation, voltage control, and energy management [7]. In this paper, we focus on the problem of voltage control. More specifically, we refer to the problem of reducing the possible overvoltage and undervoltage situations by adjusting the existing controllable devices such as static var compensators (SVCs) in the grid according to the current state of the grid, which is also called the active voltage control problem [31].

In recent decades, the deployment of various advanced communication and measurement devices in the power system, such as wide area monitoring systems (WAMS) [16], has led to continuous reduction in the cost of obtaining grid data. Traditional voltage control algorithms, such as optimal power flow [3, 13, 36], suffer from heavy calculation burden, the need for accurate physical models and a large number of manually designed parameters, making it difficult to efficiently utilize a large amount of data to adapt to today's real-time changing grid dispatch scenarios. Meanwhile, deep reinforcement learning (DRL) algorithms have demonstrated impressive performance due to their successful application in real-world scenarios such as data cooling centers [25], robotics [10, 19], and autonomous driving [26]. Therefore, data-driven and fast inference DRL algorithms for active voltage control are gaining attention [7].

Since control units are usually distributed, active voltage control problems are often solved by MARL algorithms. In prior works, researchers have improved the performance of voltage control policy learned by MARL from the perspectives of critic network design and gradient computation [5, 30], reward function design [34, 39] and so on. These methods only indirectly affect the actions given by the agents' policies and do not take into account the nonlinear relationship between actions and the bus voltage in the grid. These algorithms still have a high number of trial-and-errors to implicitly learn the effect of actions on voltage.

To address the limitations of the above algorithms, we propose a centralized safety layer-based MARL algorithm in a constrained Markov game framework, which can be combined with any continuous action MARL algorithms. Inspired by [9], We pre-train the voltage prediction network with the data from the interaction of the agents with stochastic policy. Based on the first-order approximation of this network, we construct a safety layer that corrects unsafe actions when dangerous scenarios will arise. Experiments on MAPDN [31] environment show that the safety layer helps the agents substantially improve the control rate in all grid scenarios.

Besides, we propose two components to improve the performance of the multi-agent policies that rely only on local observations during decentralized execution. The first component is inspired by the Trust Region Policy Optimization (TRPO) [27] and Proximal Policy Optimization algorithms (PPO) [28]. As we can get the corrected safe action by the safety layer, the Kullback-Leibler divergence of the action before and after correction is calculated, which can be used as a penalty term for the policy loss. The second component is inspired by [37]. We connect an action correction sub-network after the policy network and use the output of the safety layer to construct an auxiliary task to train the policy sub-network. The policy sub-network maps the unsafe actions given by the original policy network of the agents to safe actions. Experiments on MAPDN [31] demonstrate that the above two components can improve the performance of the agents' policies in some scenarios.

In summary, this paper has the following three contributions:

(1) We propose a safety layer approach in the multi-agent Constrained Markov Game framework for the active voltage control task, which effectively reduces the voltage out of the safe range.
(2) We propose the action correction penalty loss. It can be added directly to the policy loss to guide the policy closer to the safe action distribution suggested by the safety layer during training.
(3) We propose the action correction sub-network, which can efficiently exploit the local observations of the agent to correct dangerous actions instantly. The policy with the sub-network even learns safer actions than the safety layer under the guidance of the safety layer.

The remainder of this paper is organized as follows. Section 2 describes the work related to active voltage control with reinforcement learning. Section 3 will specify the active voltage control problem and model it as a multi-agent Constrained Markov Game. The methodology is described in detail in Section 4. The results of the simulation environment are presented and discussed in Section 5. Finally, we summarize our work in Section 6.

2 RELATED WORK

2.1 Safe RL for Active Voltage Control

Safe RL builds on Constrained Markov Decision Process (CMDP) [4] framework that requires the agent to maximize discounted cumulative returns while satisfying security constraints. A convenient way to implement safe RL is to perform reward function design which is equivalent to the Lagrangian multiplier method [18]. According to the prior knowledge of the grid, the reward function is carefully designed to guide the agents to maintain the bus voltage in a safe range [40]. In [35], the Constrained Policy Optimization [1] for solving the CMDP problem was directly used for Volt-Var control in the grid. They limit single-step policy updates to growth directions that do not violate constraints by means of local policy search [24] and trust region optimization [27]. Based on the core idea of Lyapunov-based policy learning [23], [29] constructs a Lyapunov function for the voltage control problem to guide the policy learning and ensure the global security of the policy during training and deployment. The above methods consider optimizing policy with gradient information related to security constraints. Our proposed safety layer can be used to correct the actions that are output by the policy and are considered dangerous during testing instead of optimizing the policy directly during training. And the above safe RL algorithms for the voltage control task are all single-agent algorithms with centralized execution. To overcome this limitation, we propose two components that utilize the output of the safety layer, and they are based on the centralized training with decentralized execution (CTDE) framework [22].

Although [17] also used the safety layer to correct the action, they need to obtain the relationship between actions and voltage by calculating the power flow equations based on the detailed physical quantities of the grid. Thus, their method requires prior knowledge of the grid topology, device parameters, and others for a specific grid, and is also based on the single-agent setting. To adapt to various complex and uncertain grid environments, we propose the safety layer that does not require any prior knowledge and is completely data-driven. Further, we discuss how to improve the policy without using the safety layer during decentralized execution.

Also relevant to our work is [9], which assumes that the dynamics of the environment are governed by first and second-order differential equations. Besides, their method only allows at most one constraint to be active and is still based on the single agent setting. In contrast, we are faced with complicated dynamics based on optimal power equations that do not satisfy simple first-order and second-order relations in active voltage control. And we also need to deal with voltage constraints that are consistent with the number of buses in the grid.

2.2 MARL for Active Voltage Control

Recently, MARL algorithms in active voltage control have received extensive attention. In [5, 33], the attention model was used in the critic network of the MADDPG algorithm to enhance scalability for more control units. [34] adopt Markov Game formulation [21] with specially designed reward for each agent. Further, [39] takes network loss and voltage violation into the design of the reward function in the multi-agent Deep Q-learning algorithm. [30] proposed a two-stage volt-var control method, in which the traditional optimal flow method [3, 13, 36] is used to dispatch on-load tap changer (OLTCs) and capacitor banks (CBs) hourly in the first stage, and only in the second stage the MADDPG algorithm is used to regulate the roof-top photovoltaics (PVs) reactive power to reduce the rapid voltage fluctuations. A consensus-based maximum entropy MARL framework is proposed by [14], which encourages the exploration of agents, but wants policies of neighboring agents to be as similar as possible in the process of training. [38] applies safe RL to MARL in energy management of microgrids, where each agent

needs to receive Lagrange multiplier variables from neighboring agents and optimize local and global constraints using primal-dual optimization method [6].

From the above work, it can be seen that as safe MARL algorithms are less studied [15], there are fewer methods to consider the application of safe MARL in active voltage control problems. Component to these research gaps, we propose the safety layer approach applied to the voltage control task in the multi-agent Constrained Markov Game framework. Moreover, we exploit the safety layer to improve the performance of policies during decentralized execution.

3 PROBLEM FORMULATION

In this section, we briefly present background information on the issue of active voltage control on power distribution networks. Subsequently, the active voltage control problem is formalized as a Multi-agent Constrained Markov Game and thus solved using the MARL algorithm.

3.1 Active Voltage Control on Power Distribution Networks

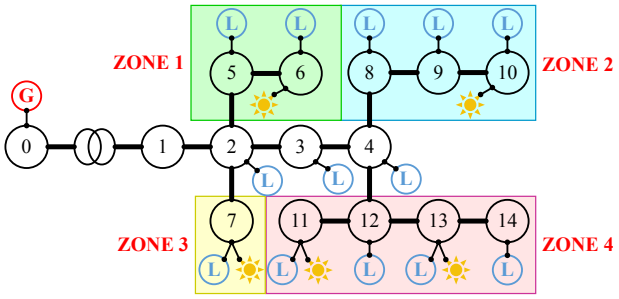


Figure 1: An example of a distribution. The black circle with a number indicates the bus, L indicates the load, and G indicates the external generator. The sun emoji indicates that there is a PV installed in this location. The network is divided into 4 parts based on the shortest path to the main branch (nodes 1 - 4). We control the voltages of bus 2 - 14, and the voltages of bus 0 and 1 are constant value v_{ref}.

In this paper, the power distribution network with distributed roof-top photovoltaics (PVs) installed is modeled as a tree graph $\mathcal{G} = (V, E)$, where $V = \{0, 1, \ldots, N\}$ and $E = \{1, 2, \ldots, N\}$ are as denoted as the set of buses (nodes) and the set of branches (edges), respectively [13]. A simple example of a power distribution network is shown in 1. It should be noted that bus 0 and bus 1 are two special buses that represent the substation or main grid with constant voltage and infinite active and reactive power capacity. They are mainly used to balance the active and reactive power of the distribution network. For each bus $i \in V \setminus \{0\}$, $s_i = p_i + jq_i$ denotes the complex power injection, where p_i is active power and q_i is reactive power, v_i and θ_i denote the magnitude and the phase angle of the complex voltage. There is a complex nonlinear relationship between these physical quantities satisfying the dynamics rules of the power system [12].

As shown in Figure 1, different control areas are divided according to the shortest path from the terminal bus to the main branch. Loads and PVs are placed on some buses. Each PV contains an inverter that maintains the voltage at each bus of the grid around the standard value denoted as v_{ref} by generating reactive power. Each PV inverter is treated as an agent and makes actions based on the observations in the area where the agent is located. For safe and stable operation of the grid, the voltage deviation needs to be kept within 5%. In other words, let the standard value $v_{\text{ref}} = 1.0$ per unit ($p.u.$), the voltage amplitude of each bus needs to satisfy $0.95\ p.u. \le v_i \le 1.05\ p.u., \forall i \in V \setminus \{0\}$. PVs convert solar energy into active power to be injected into the grid during the day, which may cause the voltage amplitude at some buses to exceed $1.05\ p.u.$. Due to heavy load at night, the end-user voltage amplitude may be less than $0.95\ p.u.$. Thus, it is necessary to suppress voltage fluctuations beyond the safe range by regulating the reactive power injection of the PV inverter.

3.2 Multi-Agent Constrained Markov Game in Active voltage Control

In this paper, we formalize the control process of the PV inverters as Multi-agent constrained Markov game, which is defined by $\langle \mathcal{N}, \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{P}, \rho_0, \gamma, C, c \rangle$. $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times S \to [0, 1]$ is probabilistic transition function, ρ_0 is initial state distribution, $\gamma \in [0, 1)$ is the discount factor, and the remaining components of the problem definition are described in detail as follows:

Agents. Each agent in $\mathcal{N} = \{1, 2, \ldots, n\}$ represents a PV. Each PV injects reactive power through the PV inverter and collaborates with each other to ensure that the voltage of each bus is within the safe range.

State and Observation. $S = \left\{\mathbf{p}^L, \mathbf{q}^L, \mathbf{p}^{PV}, \mathbf{q}^{PV}, \mathbf{v}, \theta\right\}$ defines the state set, where $\mathbf{p}^L \in (0, \infty)^{|V|}$ and $\mathbf{q}^L \in (0, \infty)^{|V|}$ denote the active and reactive power of loads, respectively; $\mathbf{p}^{PV} \in (0, \infty)^{|\mathcal{N}|}$ denotes the active power injected by PVs; $\mathbf{q}^{PV} \in (0, \infty)^{|\mathcal{N}|}$ denotes the reactive power generated by PV inverters; $\mathbf{v} \in (0, \infty)^{|V|}$ and $\theta \in (-\pi, \pi)^{|V|}$ denote voltage magnitudes and phases. The observation set is defined as $\mathcal{O} = \times_{i \in \mathcal{N}} \mathcal{O}_i$, where $\mathcal{O}_i$ is the observations of agent i. Since the grid is divided into several regions, $\mathcal{O}_i$ consists of the measures of all bus states in the region where agent i is located. It is worth noting that the measures are not exactly the same as the states, because the sensors generate measurement errors.

Action. The Cartesian product of the continuous action set $\mathcal{A}_i = \{a_i : -c \le a_i \le c, c > 0\}$ for each agent i forms the joint action set $\mathcal{A}$. The continuous action of each agent denotes the ratio of the maximum reactive power it can generate, which is given by the power system.

Reward. The reward function is defined as follows:

$$r = -\alpha \cdot l_q\left(\mathbf{q}^{PV}\right) - \frac{1}{|V|} \sum_{i \in V} l_v\left(v_i\right), \tag{1}$$

where $l_q\left(\mathbf{q}^{PV}\right) = \frac{1}{|\mathcal{N}|} \left\|\mathbf{q}^{PV}\right\|_1$ is the reactive power generation loss and $l_v(\cdot)$ is a voltage barrier function. Since it is difficult to obtain the power loss of the whole grid in practice, the simulation environment uses $l_q(\cdot)$ as an easy-to-compute power loss approximation. Barrier function $l_v(\cdot)$ describes how much the bus voltage deviates from v_{ref}, and a higher value indicates a larger deviation.

The MAPDN environment [31] provides three forms of $l_v(\cdot)$: L1-shape, L2-shape, Bowl-Shape (see Appendix A.2). [31] mention that Bowl-shape combines the advantages of L1-shape and L2-shape, so we use Bowl-shape as the form of voltage barrier function in this paper. $\alpha \in (0, 1)$ is used to balance the two losses and is set to 0.1.

Constraint. Immediate-constraint function set is defined as $C = \{C_i : \mathcal{S} \times \mathcal{A} \to \mathbb{R} \mid i \in V \setminus \{0\}\}$. In the context of active voltage, $C_i(s_t, a_t)$ denotes the voltage of bus i at time step $t+1$ after the agents take joint actions $a_t \in \mathcal{A}$ at state $s_t \in \mathcal{S}$. The constraint constants set $c = \{0.95, 1.05\}$ contains two numbers that represent the upper (overvoltage) and lower bounds(undervoltage) of the safe voltage range, respectively. At each time step t, the immediate-constraint functions must satisfy the following inequality constraint:

$$0.95 \le C_i(s_t, a_t) \le 1.05, \ \forall i \in V \setminus \{0\}. \tag{2}$$

Objective. Let the policy of agent i be $\pi_i : O_i \times \mathcal{A}_i \to [0, 1]$, then the joint policy is defined as $\pi = \times_{i \in \mathcal{N}} \pi_i$. The objective of voltage control is find the optimal joint policy π^* to maximize the expected discounted cumulative return $\mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$ while satisfying the constraint (2) as much as possible.

4 METHOD

Since the safe operation of the power distribution network is the primary task of power dispatch, our proposed method focuses on maintaining the voltage within the safe range. The results in the MAPDN environment show that the bus voltage always slowly deviates from the upper and lower boundary of the safety voltage under the control of the agents [31], rather than suddenly and significantly crossing the boundary. In order to avoid the voltage out of the range when the agents cause the voltage to be about to deviate from the boundary, we apply a small correction of actions suggested by agents. Accordingly, we propose a data-driven safety layer. In Section 4.1, we describe the principle of the safe layer. In section 4.2 and section 4.3, we discuss the utilization of action correction penalty loss and action correction sub-network to improve the performance of the policies, without using the safety layer directly.

4.1 Data-driven Safety Layer

Considering that the voltage always deviates from the boundary by a small magnitude, we would like to obtain the first-order approximation of the voltage of the actions. And then the approximation is used to solve for the small correction of the safety actions. Figure 2 shows the conceptual overview of our safety layer. It should be noted that the safety layer requires global observations of all agents. In the following, we describe our method in two parts.

4.1.1 Voltage Prediction Network. We did not design the structure of the voltage prediction network specifically for the voltage prediction task or for a specific grid environment. Instead, the common residual network structure is adopted for voltage prediction.

We concatenate observations and actions at the current time step t as input. The input is then fed sequentially into a linear layer and two residual blocks to obtain the hidden features. The residual block structure is shown in Appendix B.4, which contains two linear layers with the same hidden size. Finally, the hidden features are fed into a linear layer to predict the voltage of each bus at the next

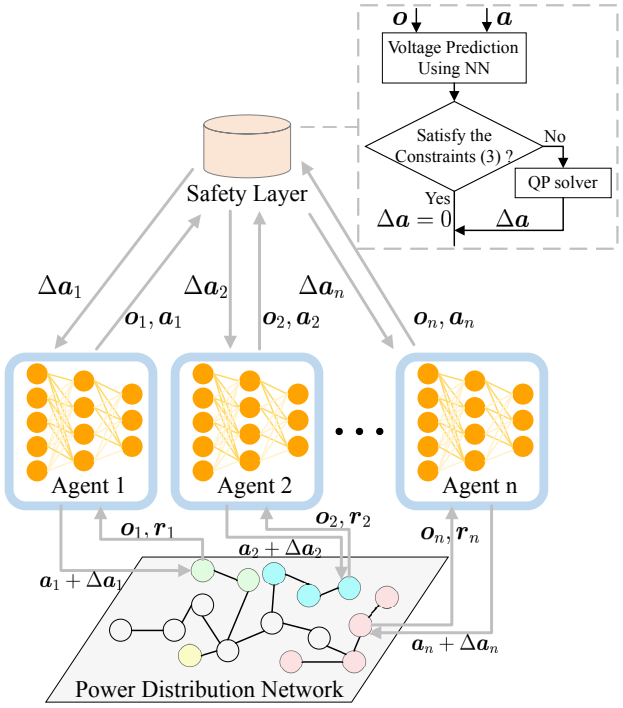


Figure 2: Schematic of MARL with the safety layer. The safety layer collects observations and actions from all agents at each time step and then predicts the voltage of each bus in the grid. If there is a bus voltage exceeding the safe range, the correction amount of actions is obtained by solving the corresponding quadratic programming problem.

time step $t+1$. The forward propagation process of the prediction network is described by the mathematical formulation as follows:

$$input^t = \text{concatenate}\left(\left(o_1^t, a_1^t\right), \left(o_2^t, a_2^t\right), \cdots, \left(o_n^t, a_n^t\right)\right), \tag{3}$$

$$feature^t = \text{Residual}\left(\text{Residual}\left(\text{Linear}\left(input^t\right)\right)\right), \tag{4}$$

$$\mathcal{V}^{t+1} = \text{Linear}\left(feature^t\right), \tag{5}$$

where o_i^t and a_i^t denote the observation obtained and the action taken by agent $i \in \mathcal{N}$ at time step t, respectively; $\mathcal{V}^{t+1}$ is the voltage vector of all buses predicted by the neural network at next time step $t+1$; further, the predicted voltage of bus $j \in V \setminus \{0\}$ at next time step $t+1$ is denoted by the notation $\mathcal{V}_j^{t+1}$.

The training process of a voltage prediction network can be considered as a supervised learning problem. $\pi' = \times_{i \in \mathcal{N}} \pi_i'$ is an arbitrary multi-agent joint policy. A transition $(o, a, r, o', done)$ can be generated after that π' interacts with the power distribution network environment at each time step, where o denotes the current observation, a denotes the action made under observation o, r denotes the reward, o' denotes the next observation and $done$ indicates whether an episode is finished. We collect these transitions as the training set for the voltage prediction network. Specifically, we take o and a in each transition as inputs to the prediction network and take the voltages of all buses in o' as the label vector v. Each sample in the training set is denoted as (o, a, v). The loss is

designed to minimize the mean square error between the predicted and actual voltage values at the next time step for all buses. The mathematical formulation of the loss function is as follows:

$$L_{\mathcal{V}}(o,a,v)=\frac{1}{K}\sum_{i=1}^{K}\frac{1}{|V|-1}\sum_{j\in V\setminus\{0\}}\left(\mathcal{V}(o_i,a_i;\theta^v)_j-v_{ij}\right)^2, \quad (6)$$

where K denotes the batch size, the subscript i denotes the i-th sample in a batch, the subscript j indexes the j-th bus in the grid environment, $V(\cdot;\theta^v)$ denotes the neural network (NN) for voltage prediction, θ^v denotes the weights in voltage prediction network.

We obtain the training set by interacting with the environment using a random joint policy that randomly and uniformly samples all actions. The network trained with the training set can be used directly in the safety layer. Our prediction network and even the safety layer do not directly affect the policy learning, i.e. our network training process and the MARL algorithm training process are decoupled from each other. Therefore, in addition to the network training method used above, we can train the prediction network while updating the actor and the critic in MARL algorithm. For example, similar to the policy update and value update process, we can train the prediction network by randomly drawing data from the replay buffer every certain interval.

4.1.2 Quadratic Programming Problem. As shown in Figure 2, when the prediction network predicts some voltages violate the constraint, the prediction network is used to model the constraint (2) as a quadratic programming problem (QP Problem) to find the safe actions. Since the voltage always deviates slowly from the boundary, dangerous cases only require fine-tuning actions to prevent it from occurring, which can be achieved by the first-order approximation of the predicted voltage.

Specifically, based on the backpropagation process of the neural network, we can obtain the Jacobi matrix of the voltage prediction values with respect to the actions at time step t:

$$\nabla_{a^t}\mathcal{V}^{t+1}=\begin{bmatrix}\frac{\partial\mathcal{V}_1^{t+1}}{\partial a_1^t} & \cdots & \frac{\partial\mathcal{V}_1^{t+1}}{\partial a_n^t}\\ \vdots & \ddots & \vdots\\ \frac{\partial\mathcal{V}_m^{t+1}}{\partial a_1^t} & \cdots & \frac{\partial\mathcal{V}_m^{t+1}}{\partial a_n^t}\end{bmatrix}_{m\times n}, \quad (7)$$

where $m=|V|-1$ denotes the number of buses controlled by the agents; $n=|\mathcal{N}|$ denotes the number of agents; $\mathcal{V}_j^{t+1}$ $(j\in V\setminus\{0\})$ denotes the predicted voltage of the j-th bus at time step $t+1$; a_i^t $(i\in\mathcal{N})$ denotes the action of the i-th agent at next time step t. According to the first-order expansion of Taylor's formula, if a small change $\Delta a^t=(a_1^t,\ldots,a_n^t)$ $(\|\Delta a^t\|\to 0)$ is added to the action a^t, the changed predicted voltage $\mathcal{V}_*^{t+1}$ has the following first-order approximation:

$$\mathcal{V}_*^{t+1}\approx\mathcal{V}^{t+1}+\nabla_{a^t}\mathcal{V}^{t+1}\Delta a^t. \quad (8)$$

If the first-order approximation of the predicted voltage is within the safe range, it is considered that the voltage of the next time step can be guaranteed to be within the safe range after the execution of the current action. In order to obtain action change amount Δa^t such that the equation (8) satisfies the voltage constraint (2), a quadratic programming problem of the following form needs to be solved:

$$\begin{aligned}&\min_{\Delta a^t}\frac{1}{2}\left\|\Delta a^t\right\|^2,\\ &\text{s.t.}\begin{cases}\mathcal{V}^{t+1}+\nabla_{a^t}\mathcal{V}^{t+1}\Delta a^t\le 1.05-\delta,\\ -V^{t+1}-\nabla_{a^t}\mathcal{V}^{t+1}\Delta a^t\le -0.95+\delta.\end{cases}\end{aligned} \quad (9)$$

where $\delta\ge 0$ is an adjustable constant for tightening the boundary. Due to the errors that exist between the first-order approximation and the actual voltage, we want to reduce the actual voltage violations as much as possible by using a more conservative bound. For the QP problem (9), we can use numerical solution methods such as interior point, active set, gradient injection and so on to solve for the best Δa^t. And then we consider $a^t+\Delta a^t$ as the safe actions that can be taken by agents in the current state.

4.2 Action Correction Penalty Loss

In this section, we consider the utilization of extra policy loss to improve the performance of the multi-agent policies with the help of the output of the safety layer. And only local observations are required for policy decentralized execution. We choose MADDPG [22] as the base algorithm to describe the training process of the whole algorithm with action correction penalty loss.

For each agent $i\in\mathcal{N}$, consider parameterizing its policy π_i with the parameter θ_i, and their joint policy is $\pi_\theta=\times_{i\in\mathcal{N}}\pi_i$, where $\theta=\{\theta_1,\theta_2,\ldots,\theta_n\}$. In MADDPG [22], the policy π_i is deterministic policy, and the gradient of the expected return $J(\pi_i)=\mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^t r_t\right]$ for agent i w.r.t. θ_i can be written as:

$$\nabla_{\theta_i}J(\pi_i)=\mathbb{E}_{o,a\sim\mathcal{D}}\left[\nabla_{\theta_i}\pi_i(a_i|o_i)\,\nabla_{a_i}Q_i^{\pi}(o,a)\Big|_{a_i=\pi_i(o_i)}\right], \quad (10)$$

where $o=(o_1,o_2,...,o_n)$ consists of the observations of all agents; $a=(a_1,a_2,...,a_n)$ consists of the actions of all agents; $\mathcal{D}$ denotes the experience replay buffer which consists of transitions $(o,a,r,o',done)$; o' denotes the next observations of all agents. The centralized action-value function Q_i^{π} is updated as follows:

$$\mathcal{L}(\theta_i)=\mathbb{E}_{o,a,r,o'\sim\mathcal{D}}\left[\left(Q_i^{\pi}(o,a)-y\right)^2\right], \quad (11)$$

$$y=r_i+\gamma\,Q_i^{\pi'}(o',a')\Big|_{a'=\pi'(o')}. \quad (12)$$

where $\pi'=\times_{i\in\mathcal{N}}\pi_i'$ is target joint policy; π_i' for each agent $i\in\mathcal{N}$ is the copy of the policy π_i network, but the network weights θ_i' of the target policy π_i slowly tracks the learned networks π_i [20].

Note that the additional information we can now obtain is the "safe" action $a^{safe}=(a_1^{safe},a_2^{safe},\ldots,a_n^{safe})$ which is generated by the safety layer corresponding to the action a. Inspired by TRPO [27] and PPO [28], we take advantage of the trust region method to make the actions learned closer to the "safe" actions. First, we expand the original quintuple $(o,a,r,o',done)$ to the hextuple $(o,a,r,o',done,a^{safe})$, i.e. we add information about the corrected safe action. Then the policy function π_i is taken to parameterize a condition Gaussian distribution $\mathcal{N}\left(a_i,\sigma_a^2\right)$. Similarly, the action of the safety layer output is parameterized as a conditional Gaussian distribution $\mathcal{N}\left(a_i^{safe},\sigma_{safe}^2\right)$. In practice, we set $\sigma_a=\sigma_{safe}=\sigma$, and σ is an adjustable hyperparameter. Finally, we construct the following constrained policy optimization problem:

$$\max_{\theta_i}J(\pi_i)\ \text{s.t.}\ \mathbb{E}_{\mathcal{D}}\left[\mathrm{KL}\left[\mathcal{N}\left(a_i,\sigma_a^2\right)\|\mathcal{N}\left(a_i^{safe},\sigma_{safe}^2\right)\right]\right]\le\delta, \quad (13)$$

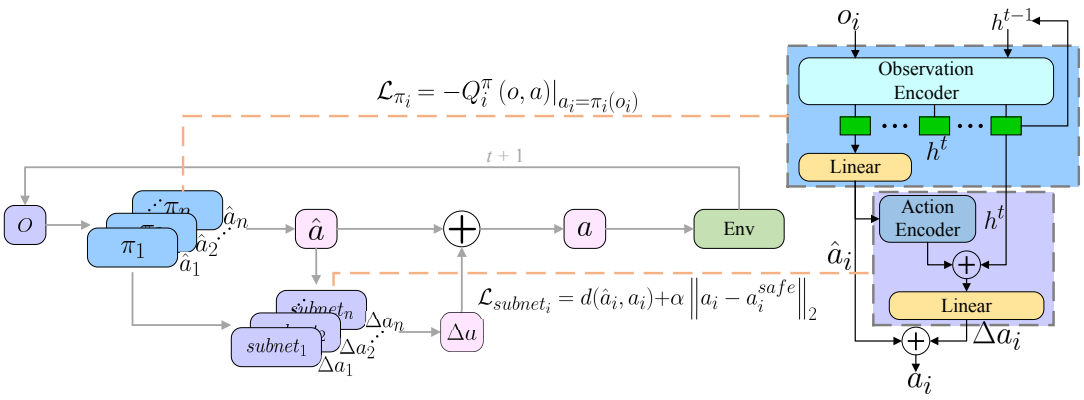


Figure 3: The diagram of action correction sub-network. The left part shows the way that the policy with sub-network interacts with the environment. The actions that are fed into the environment at each time step are original actions plus the correction amount for the actions. The right part shows the structure of the policy network and its sub-network. The observation Encoder contains the GRU module. The policy network and its sub-network are trained separately, and the two different background colors on the right part indicate the corresponding updated parameters. The formula next to the orange dashed line indicates the loss function used in the training of each network.

Similar to PPO [28], we can transform (13) into the unconstrained optimization problem with a coefficient $\beta \geq 0$ by adding a penalty (which we call action correction penalty loss) as follows:

$$\max_{\theta_i} \mathbb{E}_{\mathcal{D}} \left[Q_i^{\pi}(o,a) - \beta \mathrm{KL} \left[\mathcal{N}\left(a_i, \sigma_a^2\right) \| \mathcal{N}\left(a_i^{safe}, \sigma_{safe}^2\right) \right] \right]. \quad (14)$$

In practice, we share the parameters of the policy networks. Note that although global observations are used in the above training process, only local observations need to be entered for each agent's policy π_i during testing. The details of the MADDPG based on **action correction penalty loss (ACPL-MADDPG)** are given in Appendix B.2.

4.3 Action Correction Sub-network

The safety layer requires global observations, while the policy function of each agent receives only local observations. Thus, in the absence of the safety layer as well as global observations, another idea is that each agent directly learns the amount of change from unsafe actions to safe actions only through local observations. We add an auxiliary action correction sub-network after the policy network to learn the amount of action change. Figure 3 shows the diagram of this approach. The Observation Encoder module applies GRU [8] to map raw local observations to embeddings in the latent space. The policy network and its sub-network share hidden observation embedding. Considering that concatenating actions and observation embedding directly will increase the dimension of the sub-network input and facilitate the extension to other dimensions of action, we map the action to an embedding with the same dimension as the observation embedding. And then we add the two embeddings as the common embedding of observation and action. Finally, this embedding is mapped to the action correction amount through a linear layer.

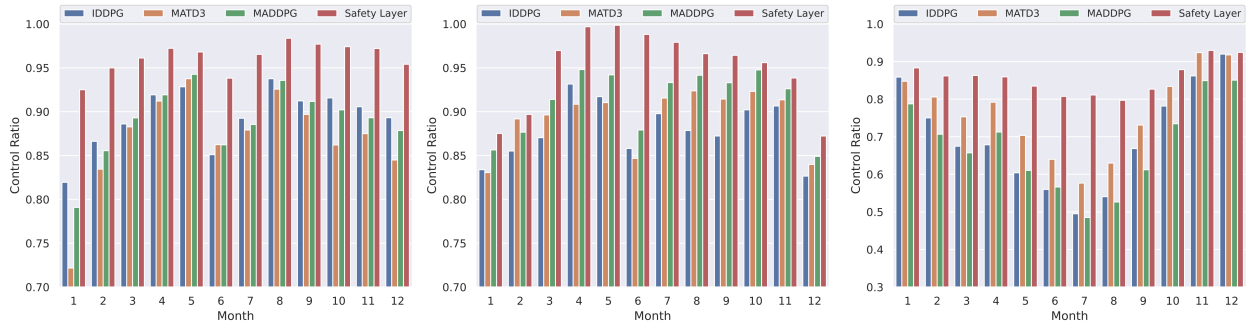
When interacting with the environment, the action a_i generated by the agent i is the original action $\hat{a}_i$ output by the policy π_i plus the action correction amount Δa_i. During training, as shown in Figure 3, the policy network and action correction sub-network will update their corresponding network parameters separately. In the same way as in section 4.2, we select the MADDPG algorithm to describe the training process of the algorithm with an action correction sub-network. Similarly, the experience replay buffer $\mathcal{D}$ consists of hextuple $(o, a, r, o', done, a^{safe})$. For each agent $i \in \mathcal{N}$, the original policy network π_i still applies the gradient of equation (10) to update its parameters. We use the following loss function to update the parameters of the action correction sub-network $subnet_i$:

$$\mathcal{L}_{subnet_i} = d(\hat{a}_i, a_i) + \alpha \cdot \left\| a_i - a_i^{safe} \right\|_2, \quad (15)$$

where the second term indicates the action correction amount to be learned by the sub-network; $\alpha > 0$ is the adjustable hyperparameter; $d(\hat{a}_i, a_i)$ is the distance function that measures the change from $\hat{a}_i$ to a_i. In section 4.1, the equation (8) used in the safety layer requires the change Δa_i of action $\hat{a}_i$ to be as small as possible; otherwise, the error of the first-order approximation will increase, leading to a decrease in the safety of the corrected action. Therefore, here we use the distance function to constrain the action correction amount learned by the sub-network not to deviate too far from the original action. In order to show the effect of action changes on returns, it was pointed out in [37] that it is more appropriate to use Q-value space distance than L2 distance in action space. And we use the distance function as follows:

$$d(\hat{a}_i, a_i) \doteq \max\left(0, Q_i^{\pi}(o, \hat{a}) - Q_i^{\pi}(o, a)\right). \quad (16)$$

When the Q-value of the corrected action a is higher than the Q-value of the original action $\hat{a}$, the loss is zero. Otherwise, the penalty is applied according to the difference in Q-values between the two actions. It should be noted that, as shown in Figure 3, only their corresponding weights are updated when the two networks are trained. The details of the MADDPG based on **action correction sub-network (ACS-MADDPG)** are given in Appendix B.3.



(a) 33-Bus (b) 141-Bus (c) 322-Bus

Figure 4: The mean test results for each month in the test dataset. "Safety Layer" indicates the test results of the policy that places the safety layer directly on the policy trained by MADDPG. The sub-caption indicates the scenario.

(a) 33-Bus (b) 141-Bus (c) 322-Bus

Figure 5: Median CR of algorithms. "ACPL-MADDPG" indicates the MADDPG algorithm with action correction penalty loss of section 4.2. "ACL-MADDPG" indicates the MADDPG algorithm with action correction sub-network of section 4.3.

5 EXPERIMENTS

5.1 Experiment Setups

5.1.1 Environment. We test the performance of the methods proposed in section 4 based on the MAPDN environment [31]. The MAPDN is a decentralized/distributed power distribution network environment for active voltage control where the MARL algorithms can be easily applied. It provides three scenarios of different scales: 33-bus network, 141-bus network, 322-bus network. The 33-bus network is divided into 4 zones and contains 32 loads and 6 PVs (agents). The 141-bus network is divided into 9 zones and contains 84 loads and 22 PVs (agents). The 322-bus network is divided into 22 zones and contains 337 loads and 38 PVs (agents). The data used for the loads and PVs are derived from real-world data of 3 years [31].

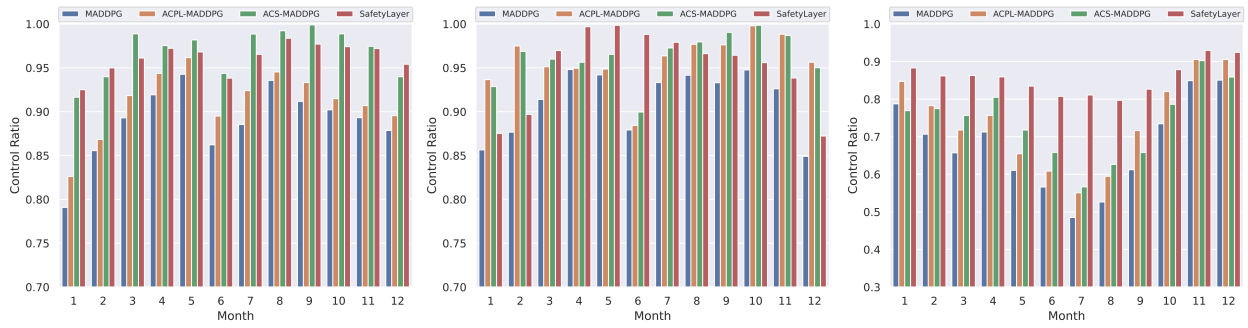
And the MAPDN provides the metric ***Control Rate (CR)*** to measure the degree of satisfying voltage constraints during the control process. More specifically, assuming that the test is conducted on M episodes and each episode lasts for T time steps, the control rate on these M episodes is calculated according to the following formula:

$$CR = \frac{1}{M}\sum_{i=1}^{M}\frac{1}{T}\sum_{t=1}^{T}\left[\mathbb{I}\left(0.95 <= Volt_t^i \text{ and } Volt_t^i <= 1.05\right)\right] \quad (17)$$

where $Volt_t^i$ is the voltage vector of all buses at time step t of the i-th episode; $\mathbf{I}(\cdot)$ is the indicator function, and its value is 1 only when all bus voltages are in the safe range.

5.1.2 Compared Methods. According to the test results of [31], MADDPG [22], MATD3 [2] and IDDPG [32] algorithms achieve excellent performance compared to other state-of-the-art MARL algorithms. We choose these three algorithms as the baselines and apply the safety layer and two components in Section 4 on the MADDPG. The voltage prediction network in the safety layer is trained using transitions generated by the interaction of the random policy with the environment, and the detailed training parameters are shown in Appendix B.4. We used the same values in [31] for the hyperparameters in the baselines, and the hyperparameters in our proposed approaches are shown in Appendix B.

5.1.3 Training and Testing. During training, each experiment is run with 5 random seeds and each episode lasts for 240 timesteps. Each experiment is evaluated on the validation dataset every 20 episodes and the evaluation results during training are given by the median and the 25%-75% quartile shading. After training is completed, we test the learned policies on the test dataset. Since



(a) 33-Bus (b) 141-Bus (c) 322-Bus

Figure 6: The mean test results for each month in the test dataset. Note that the policies of MADDPG, ACL-MADDPG, ACS-MADDPG require only local observation during execution, while the safety layer requires global observation during execution.

the difficulty of voltage control varies from month to month due to different levels of solar radiation, we test the algorithm by month. The test dataset is randomly selected from 10 days from each month, which has 120 episodes in total, and each episode lasted for 480 timesteps. The validation dataset mentioned above consists of 10 episodes randomly selected from the test dataset.

5.2 Results

Safety Layer. The test results of the MADDPG policy directly using the safety layer versus the other baselines trained policies are shown in Figure 4. In the 33-Bus, the safety layer approach achieves the control rate of about 95% in almost every month, while none of the baselines achieves that control rate. In the 141-Bus, the safety layer can still continuously improve the control rate when the baselines already have a high control rate. These verify the validity of the first-order approximation of the voltage and the small correction amount applied to the original action in Section 4.1. In the 322-Bus, the safety layer can increase the control rate substantially to about 80% even when the control rate of baselines is very low from May to August. This shows that the safety layer scales well to larger-scale grid scenarios. It can be seen that in all scenarios, the policy using the safety layer achieves the highest control rate in all months. Especially in spring and summer, excessive active power injection generated by PVs makes maintaining voltage stability more difficult. The safety layer is still able to cope well with these challenges.

ACPL and ACS. The median CR of algorithms during training in all scenarios is shown in Figure 5. Both ACPL-MADDPG and ACS-MADDPG algorithms outperform the baseline algorithms in the 33-Bus and the 141-Bus. In particular, the median control rate of the ACS-MADDPG algorithm in the 33-Bus is even able to approach 100%. In the more complex 322-Bus, the two components we propose do not have a particularly significant improvement in the training evaluation curve. This is because the number of data evaluated during training (only 10 episodes in the validation set) is too small to reflect the comprehensive performance of our components. When evaluated in the test dataset with a wider range and larger amount of data (see Figure 6(c)), the performance of our proposed components is still improved compared to the baseline in the 322-Bus. The performance of the policies trained by MADDPG, ACL-MADDPG, ACS-MADDPG, and the MADDPG policy with the safety layer in the test dataset is shown in Figure 6. In the 33-Bus and the 141-Bus, both components are effective in improving control rates compared to MADDPG, and can even achieve better performance than the MADDPG policy with the safety layer. ACPL and ACS try to guide policy closer to safe action distribution suggested by safety layer, and intuitively the upper limit of policy will not be higher than safety layer. Since agents also actively explores safer states during training (because the reward function includes the penalty item related to the voltage exceeding the safe range), the guidance of the two components and the exploration of the agent itself promote each other and form a virtuous cycle, which leads to the experimental results in the 33-Bus and the 141-Bus in Figure 6. In the 322-Bus, both algorithms do not achieve the performance of the safety layer, but compared with the MADDPG algorithm as the baseline, the control rate is improved almost every month. This is because the 322-Bus is more complex and the exploration capability of agents is limited. These results show that our proposed two components can get good performance improvement when only local observations are obtained and policies are executed decentrally.

6 CONCLUSION

In this paper, we focus on maintaining the voltage at each bus in the grid within a safe range and propose the safety layer method. Experimental results show that the policy with the safety layer approach effectively improves the voltage control rate in all scenarios. Then, considering that the agent has to make decisions relying only on local observations without the involvement of the safety layer during decentralized policy execution, we proposed two novel components: action correction penalty loss and action correction sub-network. The two components make the policy output closer to the safe action distribution of the safety layer and can even help the agent to actively learn the policy that is safer than the safety layer under the guidance of the safety layer. Experimental results show the effectiveness of our two components. In the future, we consider the utilization of larger models and the load pattern information to enable them to learn more complex cooperative strategies.

ACKNOWLEDGMENTS

This work was supported in part by NSFC under Contract 61836011, and in part by the Fundamental Research Funds for the Central Universities under contract WK3490000007.

REFERENCES

[1] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. 2017. Constrained policy optimization. In *International conference on machine learning*. PMLR, 22–31.

[2] Johannes Ackermann, Volker Gabler, Takayuki Osa, and Masashi Sugiyama. 2019. Reducing overestimation bias in multi-agent domains using double centralized critics. *arXiv preprint arXiv:1910.01465* (2019).

[3] Yashodhan P Agalgaonkar, Bikash C Pal, and Rabih A Jabr. 2013. Distribution voltage control considering the impact of PV generation on tap changers and autonomous regulators. *IEEE Transactions on Power Systems* 29, 1 (2013), 182–192.

[4] Eitan Altman. 1999. *Constrained Markov decision processes: stochastic modeling*. Routledge.

[5] Di Cao, Weihao Hu, Junbo Zhao, Qi Huang, Zhe Chen, and Frede Blaabjerg. 2020. A multi-agent deep reinforcement learning based voltage regulation using coordinated PV inverters. *IEEE Transactions on Power Systems* 35, 5 (2020), 4120–4123.

[6] Tsung-Hui Chang, Angelia Nedić, and Anna Scaglione. 2014. Distributed constrained optimization by consensus-based primal-dual perturbation method. *IEEE Trans. Automat. Control* 59, 6 (2014), 1524–1538.

[7] Xin Chen, Guannan Qu, Yujie Tang, Steven Low, and Na Li. 2021. Reinforcement learning for decision-making and control in power systems: Tutorial, review, and vision. *arXiv* (2021).

[8] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* (2014).

[9] Gal Dalal, Krishnamurthy Dvijotham, Matej Vecerik, Todd Hester, Cosmin Paduraru, and Yuval Tassa. 2018. Safe exploration in continuous action spaces. *arXiv preprint arXiv:1801.08757* (2018).

[10] Christian Schroeder de Witt, Bei Peng, Pierre-Alexandre Kamienny, Philip Torr, Wendelin Böhmer, and Shimon Whiteson. 2020. Deep multi-agent reinforcement learning for decentralized continuous cooperative control. *arXiv preprint arXiv:2003.06709* (2020).

[11] Reid Detchon and Richenda Leeuwen. 2014. Policy: Bring sustainable energy to the developing world. *Nature* 508 (04 2014), 309–11. https://doi.org/10.1038/508309a

[12] Masoud Farivar, Lijun Chen, and Steven Low. 2013. Equilibrium and dynamics of local voltage control in distribution systems. In *52nd IEEE Conference on Decision and Control*. IEEE, 4329–4334.

[13] Lingwen Gan, Na Li, Ufuk Topcu, and Steven H Low. 2013. Optimal power flow in tree networks. In *52nd IEEE Conference on Decision and Control*. IEEE, 2313–2318.

[14] Yuanqi Gao, Wei Wang, and Nanpeng Yu. 2021. Consensus multi-agent reinforcement learning for volt-var control in power distribution networks. *IEEE Transactions on Smart Grid* 12, 4 (2021), 3594–3604.

[15] Shangding Gu, Long Yang, Yali Du, Guang Chen, Florian Walter, Jun Wang, Yaodong Yang, and Alois Knoll. 2022. A Review of Safe Reinforcement Learning: Methods, Theory and Applications. *arXiv preprint arXiv:2205.10330* (2022).

[16] Vehbi C Gungor, Dilan Sahin, Taskin Kocak, Salih Ergut, Concettina Buccella, Carlo Cecati, and Gerhard P Hancke. 2011. Smart grid technologies: Communication technologies and standards. *IEEE transactions on Industrial informatics* 7, 4 (2011), 529–539.

[17] Peng Kou, Deliang Liang, Chen Wang, Zihao Wu, and Lin Gao. 2020. Safe deep reinforcement learning-based constrained optimal control scheme for active distribution networks. *Applied energy* 264 (2020), 114772.

[18] Hoang Le, Cameron Voloshin, and Yisong Yue. 2019. Batch policy learning under constraints. In *International Conference on Machine Learning*. PMLR, 3703–3712.

[19] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. 2016. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research* 17, 1 (2016), 1334–1373.

[20] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2015. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971* (2015).

[21] Michael L Littman. 1994. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*. Elsevier, 157–163.

[22] Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. 2017. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems* 30 (2017).

[23] Theodore J Perkins and Andrew G Barto. 2002. Lyapunov design for safe reinforcement learning. *Journal of Machine Learning Research* 3, Dec (2002), 803–832.

[24] Jan Peters and Stefan Schaal. 2008. Reinforcement learning of motor skills with policy gradients. *Neural networks* 21, 4 (2008), 682–697.

[25] Jim Gao Richard Evans. 2016. DeepMind AI Reduces Google Data Centre Cooling Bill by 40%. blog. Retrieved October 15, 2022 from https://www.deepmind.com/blog/deepmind-ai-reduces-google-data-centre-cooling-bill-by-40

[26] Ahmad EL Sallab, Mohammed Abdou, Etienne Perot, and Senthil Yogamani. 2017. Deep reinforcement learning framework for autonomous driving. *Electronic Imaging* 2017, 19 (2017), 70–76.

[27] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. 2015. Trust region policy optimization. In *International conference on machine learning*. PMLR, 1889–1897.

[28] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).

[29] Yuanyuan Shi, Guannan Qu, Steven Low, Anima Anandkumar, and Adam Wierman. 2022. Stability constrained reinforcement learning for real-time voltage control. In *2022 American Control Conference (ACC)*. IEEE, 2715–2721.

[30] Xianzhuo Sun and Jing Qiu. 2021. Two-stage volt/var control in active distribution networks with multi-agent deep reinforcement learning method. *IEEE Transactions on Smart Grid* 12, 4 (2021), 2903–2912.

[31] Jianhong Wang, Wangkun Xu, Yunjie Gu, Wenbin Song, and Tim C Green. 2021. Multi-agent reinforcement learning for active voltage control on power distribution networks. *Advances in Neural Information Processing Systems* 34 (2021), 3271–3284.

[32] Jianhong Wang, Yuan Zhang, Tae-Kyun Kim, and Yunjie Gu. 2020. Shapley Q-value: A local reward approach to solve global reward games. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 7285–7292.

[33] Minrui Wang, Mingxiao Feng, Wengang Zhou, and Houqiang Li. 2022. Stabilizing Voltage in Power Distribution Networks via Multi-Agent Reinforcement Learning with Transformer. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (Washington DC, USA) *(KDD '22)*. Association for Computing Machinery, New York, NY, USA, 1899–1909. https://doi.org/10.1145/3534678.3539480

[34] Shengyi Wang, Jiajun Duan, Di Shi, Chunlei Xu, Haifeng Li, Ruisheng Diao, and Zhiwei Wang. 2020. A data-driven multi-agent autonomous voltage control framework using deep reinforcement learning. *IEEE Transactions on Power Systems* 35, 6 (2020), 4644–4654.

[35] Wei Wang, Nanpeng Yu, Jie Shi, and Yuanqi Gao. 2019. Volt-VAR Control in Power Distribution Systems with Deep Reinforcement Learning. In *2019 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*. 1–7. https://doi.org/10.1109/SmartGridComm.2019.8909741

[36] Yan Xu, Zhao Yang Dong, Rui Zhang, and David J Hill. 2017. Multi-timescale coordinated voltage/var control of high renewable-penetrated distribution systems. *IEEE Transactions on Power Systems* 32, 6 (2017), 4398–4408.

[37] Haonan Yu, Wei Xu, and Haichao Zhang. 2022. Towards Safe Reinforcement Learning with a Safety Editor Policy. *arXiv preprint arXiv:2201.12427* (2022).

[38] Qianzhi Zhang, Kaveh Dehghanpour, Zhaoyu Wang, Feng Qiu, and Dongbo Zhao. 2020. Multi-agent safe policy learning for power management of networked microgrids. *IEEE Transactions on Smart Grid* 12, 2 (2020), 1048–1062.

[39] Ying Zhang, Xinan Wang, Jianhui Wang, and Yingchen Zhang. 2020. Deep reinforcement learning based volt-var optimization in smart distribution systems. *IEEE Transactions on Smart Grid* 12, 1 (2020), 361–371.

[40] Ying Zhang, Xinan Wang, Jianhui Wang, and Yingchen Zhang. 2021. Deep Reinforcement Learning Based Volt-VAR Optimization in Smart Distribution Systems. *IEEE Transactions on Smart Grid* 12, 1 (2021), 361–371. https://doi.org/10.1109/TSG.2020.3010130