2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

MAPPG: Multi-agent Phasic Policy Gradient

Qi Zhang[1], Xuetao Zhang[1⋆], *IEEE Member*, Yisha Liu[2], *IEEE Member*,
Xuebo Zhang[3], *IEEE Senior Member*, Yan Zhuang[1] *IEEE Senior Member*

***Abstract*— We propose a Multi-Agent Phasic Policy Gradient (MAPPG) algorithm, which can assist agents to further alleviate the non-stationarity of the environment. Different from the existing methods, the auxiliary phase is introduced to train the local policy directly by using the environment state, which can be naturally integrated into other algorithms. Specifically, the hidden layer feature sharing is proposed, which ensures feature sharing between the global value network and the local policy network for the first time. Meanwhile, mirror descent is utilized to iteratively update the policy in the auxiliary stage, which makes the policy update more robust. Through a series of evaluations on multi-agent Particle and multi-agent Mujoco benchmark environments, the experimental results show that our method achieves higher rewards than state-of-the-art benchmarks.**

I. INTRODUCTION

Multi-agent reinforcement learning (MARL) has attracted much attention in recent years, which can be applied in collaborative tasks, such as wave energy converters [1] and robot swarm control [2]. As for scalability and communication limitation problems, the existing methods mainly leverage the decentralized execution of multi-agent policies that act only on their local observations. The most intuitive decentralized approach is to treat other agents as part of the environment, such as Independent synchronous Advantage Actor-Critic (IA2C) [3] and Independent Proximal Policy Optimization (IPPO) [4] algorithms, which perform well for many tasks. However, due to the problem of environmental non-stationarity in some partially observable tasks, the performance still needs to be improved. Therefore, Centralized Training and Decentralized Execution (CTDE) [5] is proposed to mitigate the non-stationarity of the environment by using additional information during training.

Strong benchmarks such as Multi-agent PPO (MAPPO) [6] and QMIX [7] are based on the framework of CTDE, which utilize the environment state to train the global state-action function or value function during the training phase. Then, the value function is exploited to guide the training of local policies. In other words, the above CTDE framework can be regarded as a credit assignment mechanism, however, it is still an indirect way. Naturally, it is promising to train local policies directly using environment states, which may further mitigate the non-stationarity of the environment. However, environment states can not be integrated directly since the CTDE framework needs to ensure that policy execution relies only on its own local observations.

To maintain the independence between the policy execution and the environment state, it is necessary to ensure that the state is only utilized for the training network, which means it is only operated for the auxiliary training. Motivated by single-agent algorithm Phasic Policy Gradient (PPG) [8], the feature sharing between critic network and actor network is without mutual interference of update targets. However, the above feature sharing is difficult to achieve in multi-agent tasks, because the dimension gap between state and observation is large and there is no theoretical guarantee to realize feature sharing in the auxiliary phase.

Motivated by these observations, we propose a MAPPG algorithm, which trains the local policy directly by using the environment state, assisting agents to further alleviate the non-stationarity of the environment. In detail, independent networks are utilized to express the value function and policy to avoid the influence between the optimization objectives. Moreover, an auxiliary phase is introduced to realize that the policy network learns useful features from the value network. To conclude, our contributions are summarized as follows:

- We propose a MAPPG algorithm to integrate the hidden layer feature sharing to ensure feature sharing between the value network and the policy network, which can further mitigate the non-stationarity of the environment. It should be noted that the proposed framework can be applied to all the CTDE systems.
- An additional auxiliary phase is introduced to update the policy, and the general mirror descent method is utilized to formulate the auxiliary policy update into a trust region optimization problem, which can improve the robustness of the update.
- The experimental results show that our proposed method has obtained the higher average rewards in multi-agent particle (MPE) and multi-agent MuJoCo (MAMuJoCo) environments, surpassing the benchmarks of MAPPO and MADDPG.

This work was supported in part by the National Natural Science Foundation of China under 62103077, 61973049, and U1913201, in part by Natural Science Foundation of Liaoning Province under Grant 2022-KF-12-05. *(Corresponding author: Xuetao Zhang.)*

[1]Qi Zhang, Xuetao Zhang, Yan Zhuang are with the Intelligent Robotic Laboratory, School of Artificial Intelligence, also School of Control Science and Engineering, Dalian University of Technology, Dalian 116024, China. kxcqizhang@mail.dlut.edu.cn, zhangxuetao@dlut.edu.cn, zhuang@dlut.edu.cn

[2]Yisha Liu is with the School of Information Science and Technology, Dalian Maritime University, Dalian, China, 116026. liuyisha@dlmu.edu.cn

[3]Xuebo Zhang is with the College of Artificial Intelligence, the Institute of Robotics and Automatic Information System (IRAIS), Tianjin Key Laboratory of Intelligent Robotics, Nankai University, Tianjin 300350, China. zhangxuebo@nankai.edu.cn

II. Related Work

It has always been the development trend of multi-agent algorithms to extend the existing single-agent algorithm to the multi-agent domain, which is mainly divided into the value decomposition method [7], [9], [10] and the multi-agent policy gradient (MAPG) method [3], [6], [11].

The value decomposition method can actually be regarded as combining the value functions in several single-agent algorithms into global value function through functional relationships, to obtain the overall information and ensure scalability. For example, VDN [9] is the sum of local Q-functions of several DQN networks to represent the global Q-function, QMIX [7] is to add mixed networks to ensure monotonicity constraints, and FACMAC [10] is to replace the underlying DQN with DDPG [12]. However, it is faced with the problem of relative over generalization, which converges to the local optimal solution incorrectly. In contrast, the MAPG method does not add monotonicity constraint, and its expression is not restricted.

Due to the theoretical guarantee of multi-agent policy gradient theorem [13] and excellent effect in continuous action space, MAPG algorithm has been a research hotspot. Particularly, MADDPG [11] extends DDPG algorithm by adding inference to other agent policies, MAPPO develops PPO [14] algorithm by using techniques such as value function standardization and removal of overlapping features, and HAPPO [15] inherits PPO algorithm by employing multi-agent advantage decomposition lemma and the sequential policy update scheme. However, the above algorithms use environmental states to train the global value function and then guide local policies through the value function, not in a direct way. To train local policies directly using environmental states, the proposed MAPPG algorithm extends the PPG algorithm which designs a hidden layer feature sharing mechanism to solve the problem of dimensionality inconsistency, achieving the higher average rewards in both MPE [16] and MAMuJoCo [17] environments.

The technique of parameter sharing is widely applied in multi-agent algorithms [6], [7], [9], [18]. For instance, the MAPPO method uses agent sequence number as a distinguishing way to share parameters of all local policies. In general, the most commonly approach is to fully share model parameters among agents. However, to the best of our knowledge, research on the sharing between global value functions and policy networks in multi-agents is rarely done. Our proposed MAPPG method can remedy this defect, which is the first feature sharing mechanism between the global value network and the local agent policy network.

III. BACKGROUND

This work is dedicated to solving a fully cooperative multi-agent task. It can be typically formulated as a decentralized partially observable markov decision process (Dec-POMDP) [19], which is expressed as a tuple $G = \langle N, S, \boldsymbol{U}, P, r, Z, O, \gamma, d\rangle$, where $N = \{1, ..., n\}$ denotes the set of agents, S is the finite state space, $\gamma \in [0, 1)$ represents the discount factor and d stands for the initial state distribution. At each time step t, each agent $i \in N$ obtains its a partial observation o_t^i according to the observation function $O(s_t, i) : S \times N \to Z$ and takes actions $a_t^i \in U^i$ according to its policy $\pi^i(\cdot|o_t^i)$. The actions of all agents form a joint action $\boldsymbol{a}_t = (a_t^1, ..., a_t^n) \in \boldsymbol{U}$ to interact with the environment. Then, the next state is obtained according to the state transition function $P(s_t'|s_t, \boldsymbol{a}_t)$, and the reward is obtained according to the shared reward function $r(s_t, \boldsymbol{a}_t)$. For convenience, we define the state distribution as $\rho_{\boldsymbol{\pi}} \triangleq \sum_{t=0}^{\infty} \gamma^t Pr(s_t = s|d, \boldsymbol{\pi})$, and the goal is to maximize the expected discount reward:

$$J(\boldsymbol{\pi}) = E_{s\sim\rho_{\boldsymbol{\pi}}, \boldsymbol{a}\sim\boldsymbol{\pi}}[\sum_{t=0}^{\infty} \gamma^t r(s_t, \boldsymbol{a}_t)] \tag{1}$$

A. Mirror descent

Mirror descent (MD) is a first-order confidence region optimization method to solve constrained optimization problems. MD update can be expressed as the following formula

$$x_{k+1} = \arg\min_{x\in C} \langle \nabla f(x_k), x - x_k\rangle + \frac{1}{t_k} B_\psi(x, x_k) \tag{2}$$

where f is a convex function, the constraint set C is convex compact, $B_\psi(x, x_k) \triangleq \psi(x) - \psi(x_k) - \langle \nabla\psi(x_k), x - x_k\rangle$ is the Bregman divergence related to the strongly convex function ψ, and t_k represents a step-size [20]. It should be emphasized that when ψ is negative Shannon entropy, $B_\psi(x, x_k)$ becomes the form of KL divergence.

B. Phasic Policy Gradient

PPG improves PPO algorithm of shared network to reduce interference between policy and value function objectives, while still sharing representation, enabling policy network to learn useful features from value function network in the auxiliary phase [8].

$$L^{joint} = L^{aux} + \beta_{clone} \cdot \hat{E}_t[KL(\pi_{\theta_{old}}(\cdot|s_t), \pi_\theta(\cdot|s_t))] \tag{3}$$

where L^{aux} stands for auxiliary optimization objective, $\pi_{\theta_{old}}$ is the policy before the auxiliary phase begins, β_{clone} is a fixed hyperparameter that controls the relative importance of the original policy and the auxiliary objective.

IV. ALGORITHMS

All illustrate in Fig. 1, we focus on the hidden layer feature sharing mechanism to deal with the problem of alternating input of environmental state and local observations. Then the mirror descent is introduced as the theoretical basis of the auxiliary stage, and the derivation of the actual algorithm accordingly.

A. Network Framework

In the multi-agent PG algorithm that conforms to the CTDE framework, the global value network employs the state s as the input for training, and the actor network of each agent almost utilizes the local observation o for learning. Due to the influence of partial observability under multi-agent tasks, the dimension of s is generally much larger than that of o. In general, s can be regarded as the concatenation of

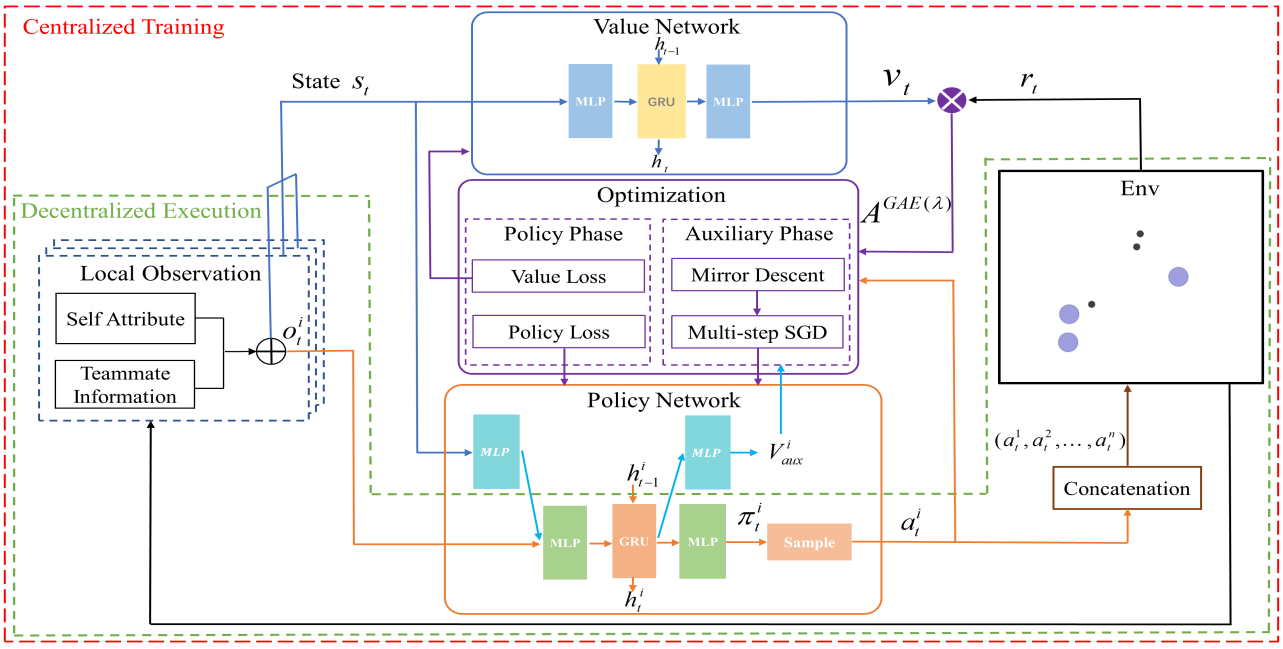


Fig. 1. The framework of MAPPG. The training of policy network is divided into two alternating phases: the policy phase and the auxiliary phase. In addition, the value head of the policy network is only utilized to give training objectives in the auxiliary phase.

observation o by various agents. If the policy network also needs state s for auxiliary training, the primary problem is that the dimensions of state and observation are inconsistent.

To solve the problem of inconsistent dimension, we propose a MAPPG algorithm framework, whose policy network integrates the hidden layer sharing mechanism. First, the policy network of MAPPG algorithm only shares the hidden layer, which avoids the problem of dimensionality inconsistency and ensures the feature sharing between the value network and the policy network. Second, the shared part of the policy network includes the GRU network, whose hidden state utilizes the hidden state of the agent itself. This is because the purpose of auxiliary training is to benefit the policy network, which is equivalent to using the global state to guide the policy training in each step.

B. Policy Phase

Compared with the MAPPO algorithm, the contribution of the proposed MAPPG algorithm is to introduce an additional auxiliary phase. During the policy phase, the optimization objective is consistent with MAPPO, and different networks are utilized to represent the policy and value function to reduce the interference between the update objectives. Specifically, the policy network is trained using the clipped surrogate objective:

$$E_{s\sim\rho_{\boldsymbol{\pi}},\boldsymbol{a}\sim\boldsymbol{\pi}_{\theta_k}}[\min(r^i(\theta)\hat{A}(s,\boldsymbol{a}),\text{clip}(r^i(\theta),1\pm\epsilon)\hat{A}(s,\boldsymbol{a}))] \tag{4}$$

where $r^i(\theta)=\frac{\pi_\theta^i(a^i|s)}{\pi_{\theta_k}^i(a^i|s)}$, $\hat{A}$ is an estimator of the advantage function. To encourage the exploration of agents, the final optimization goal is to add a local policy entropy bonus $S[\pi_\theta^i]$ to (4), and operate parameter β_S to trade off the importance between them.

To train the global value function, we need to optimize the objective of

$$L_{value}=E_{s\sim\rho_{\boldsymbol{\pi}}}[(V_\phi(s)-\hat{R})^2] \tag{5}$$

where $\hat{R}$ is the discounted reward-to-go, θ and ϕ denote the parameters of policy network and value network respectively. Both $\hat{A}$ and $\hat{R}$ are estimated by GAE [21].

C. Auxiliary Phase

During the auxiliary phase, the update of the policy network should not change its policy as much as possible while approaching the optimization objective, which can be regarded as a constrained optimization problem in essence. Since the policy represents a probability distribution, it is natural to utilize KL divergence to measure the degree of policy change. Therefore, the joint optimization objective is

$$\theta_{k+1}=\arg\min_\theta L(\theta,\theta_k) \quad where \tag{6}$$

$$L(\theta,\theta_k)=E_{s\sim\rho_{\boldsymbol{\pi}}}[(V_\theta^i(s)-\hat{R})^2+\frac{1}{t_k}KL(\pi_\theta^i(\cdot|s),\pi_{\theta_k}^i(\cdot|s))]$$

To learn useful features from the value network, the updated objective of the local policy network is the same as that of the value network, that is, $\hat{R}$ utilized here is the same as (5). Notice that the direction of KL is different from the direction of PPG, $KL(\pi_{\theta_k},\pi_\theta)$. Moreover, instead of using a fixed parameter β_{clone} to control the trade-off between the auxiliary optimization goal and the original policy, an annealed schedule is employed to dynamically update parameter t_k, which slowly decreases from 1 to near 0. In the initial training stage, it is known from (6) that useful features can be obtained from the value function with comprehensive information to a greater extent. However, in the training convergence stage, more emphasis is placed on retaining

its policy, which prevents all local policies from becoming resemblance.

In a word, (3) and (6) are distinguished by the direction of KL divergence and an annealed schedule. Different from (6), PPG algorithm only empirically determines the way to update the policy through (3) in the auxiliary stage, without theoretical guarantee. If the Bregman divergence term takes the form of the KL divergence, i.e., $B_\psi(x, x_k) = KL(x, x_k)$, then it is easy to see that (6) conforms to the general MD updating rules in (2). As a result, it provides a theoretical basis for policy updating in the auxiliary stage, which brings the improvement of convergence and robustness. It is worth noting that the auxiliary phase is controlled by a hyperparameter K_{aux}, which is not executed for every update. Therefore, it can be seen as performing updates on different periods, which is why updates are divided into two alternating phases.

Moreover, if updating (6) in the auxiliary phase by performing a single stochastic gradient descent (SGD) step as

$$\nabla_\theta KL(\pi_\theta^i(\cdot|s), \pi_{\theta_k}^i(\cdot|s))|_{\theta=\theta_k} = 0 \tag{7}$$

and thus the gradient of (6) is the following equation

$$\nabla_\theta L(\theta, \theta_k)|_{\theta=\theta_k} = \nabla_\theta E_{s\sim\rho_{\boldsymbol{\pi}}}[(V_\theta^i(s) - \hat{R})^2] \tag{8}$$

the resulting algorithm would be equivalent to unconstrained optimization, which misses the purpose of preserving the original policy, making the local policies very similar. As a consequence, m SGD steps need to be performed at each iteration k as follows

$$\begin{aligned} \theta_k^{(i+1)} &= \theta_k^{(i)} + \eta\nabla_\theta L(\theta, \theta_k)|_{\theta=\theta_k^{(i)}} \\ where \quad \theta_k^{(0)} &= \theta_k, \theta_{k+1} = \theta_k^{(m)} \\ for \quad i &= 0, ..., m-1 \end{aligned} \tag{9}$$

In the actual algorithm, the update of the auxiliary stage is according to (9), except that observation o^i is utilized in KL to replace the state s, i.e., $KL(\pi_\theta^i(\cdot|o^i), \pi_{\theta_k}^i(\cdot|o^i))$.

In a typical implementation of MAPPO, value function optimization and policy optimization utilize the same level of sample reuse, making it difficult to mediate the relationship between policy and value function optimization. In contrast, the MAPPG algorithm allows the value function to be trained several additional times in the auxiliary phase. Therefore, the value function optimization and policy optimization can be decoupled to a certain extent, resulting in a higher level of sample reuse, which helps to train more accurate value functions.

V. EXPERIMENTS

In this section, Multi-agent Particle Environment (MPE) with discrete action space and Multi-Agent MuJoCo (MA-MuJoCo) with continuous action space are selected for more comprehensive algorithm performance testing. The proposed MAPPG is compared with the state-of-the-art benchmarks including MADDPG and MAPPO in these two environments with different characteristics.

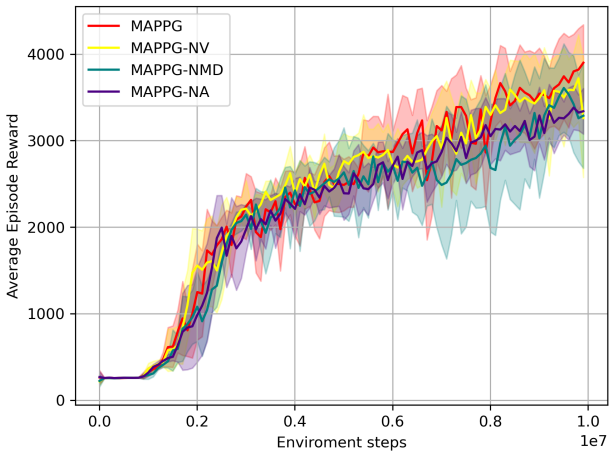


Fig. 2. Effect of different settings of MAPPG algorithm on the performance of Walker 2×3 in the auxiliary phase. The suffix NV in the label means that no additional updates are made to the value network in the auxiliary phase, the suffix NMD means that no mirror descent is used to update the policy network, and NA implies that neither of the above is utilized.

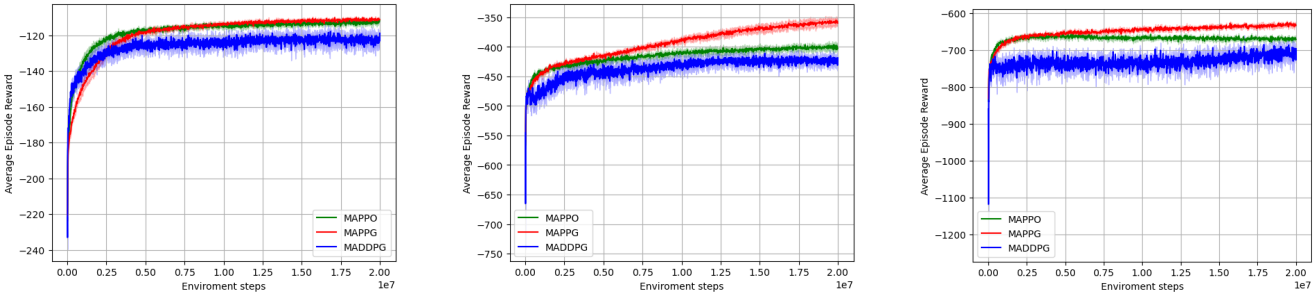
Experimental Setup: Each experiment was conducted on a laptop computer with 16 GB of RAM, one 14-core CPU, and one GeForce RTX 3070Ti GPU, which is utilized for network training and environment interaction. The MPE environment used in the experiment is consistent with that in paper [6], and the MAMujoco environment follows the specification in paper [15]. For a fair comparison, the MAPPO benchmark employs open-source code in the original paper [6], and the implementation of MADDPG follows the EPyMARL [3] framework. It is noted that the experimental parameters are all adopted from the original paper or its released code parameters.

Evaluation Method: In all the tables given in this experiment, the highest value in each task is presented in bold, and the asterisk indicates the algorithm that is not significantly different from the best execution algorithm in each task. In addition, each algorithm evaluates the performance of five random seeds for each task. The solid line in the figure represents the average set reward for each algorithm in the five seed evaluations, and the shadowed part represents the 95% confidence interval.

A. Ablation Studies

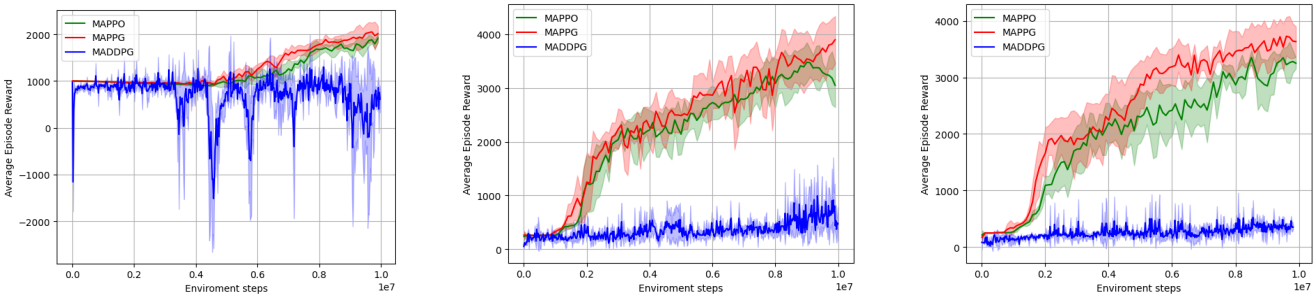
Ablation experiments are conducted to investigate the effect of additional updating of the value network and MD method in the auxiliary phase on the performance of the algorithm. Therefore, four ablation experiments can be constructed according to whether there is an additional update value network or an MD method.

Fig. 2 shows the results of these ablation experiments on MAMuJoCo. It can be clearly seen that the MD method can increase reward and robustness in the training process compared to the updated method of (3). In addition, adding additional value network updates in the auxiliary phase can help further increase the reward. Therefore, the proposed MAPPG algorithm has the best performance.



(a) Spread (3 agents) (b) Spread (6 agents) (c) Spread (8 agents)

Fig. 3. Performance comparison on Spread scenario of MPEs with different number of agents. The shadowed part represents the 95% confidence interval. The solid line represents the average episode reward of each algorithm in the evaluation of the five seeds.



(a) Ant 2×4 (b) Walker 2×3 (c) Walker 6×1

Fig. 4. Performance comparison of three typical Multi-Agent MuJoCo tasks. In this experiment, the technique of parameter sharing is not utilized, which greatly improves the performance and facilitates the comparison of the real difference between algorithms.

TABLE I

AVERAGE EPISODE REWARD OF THE LAST 10 EVALUATIONS OF EACH ALGORITHM ON 5 SEEDS IN THE MPES.

Tasks\Algs	MADDPG	MAPPO	MAPPG
Spread(3 agents)	-122.87	-112.47*	**-111.19**
Spread(6 agents)	-424.71	-400.81	**-358.10**
Spread(8 agents)	-706.08	-670.02	**-632.76**
Speaker-Listener	-27.67	**-12.13**	-12.64*
Reference	-13.66	-9.78*	**-9.68**
Avg.	-259.00	-241.04	**-224.87**

TABLE II

AVERAGE EPISODE REWARD OF THE LAST 10 EVALUATIONS OF EACH ALGORITHM ON 5 SEEDS IN THE MAMUJOCO.

Tasks\Algs	MADDPG	MAPPO	MAPPG
Ant 2×4	661.27	1781.41	**1967.86**
Walker 2×3	628.94	3310.29	**3674.71**
Walker 6×1	359.70	3186.88	**3616.86**
HalfCheetah 2×3	3106.31	4447.16*	**4501.67**
HalfCheetah 6×1	4179.61	4433.01*	**4444.17**
Avg.	1787.17	3431.75	**3641.05**

B. MPE Results

MPE is a 2-dimensional multi-agent particle world with continuous observation space and discrete action space, including some predator-prey, navigation, and communication tasks. In this experiment, we compare the MAPPG algorithm with the MAPPO and MADDPG algorithms in focusing on three cooperative tasks: Speaker-Listener, Spread, and Reference. In the Spread task, each agent needs to control a point to navigate to landmarks, which has variable agents and target points so that it can be evaluated in more detail. Moreover, all algorithms adopt the local policy parameter sharing technique to improve the computational efficiency in MPEs. The convergence performance of each algorithm is shown in Fig. 3.

The more comprehensive results are listed in Table I, which demonstrates that MADDPG achieves the worst results in all of the tasks. In contrast, the MAPPG algorithm performs optimal performance in all five tasks. Especially under the Spread task, MAPPG is more robust as the number of agents and task complexity increase. For typical tasks such as Reference, Speaker-Listener and Spread that contain only 2 or 3 agents, the number of agents is small, and the MAPPG algorithm is similar to the MAPPO algorithm. Taking the Spread task as an example, Fig. 3(b) reveals that when the number of agents increased to six, the MAPPG algorithm significantly exceeds the other two methods. In addition, as shown in Fig. 3(c), when the number of agents increases to 8, the performance of MAPPG is more robust, while the reward of MAPPO tends to decline with the training.

C. MAMuJoCo Results

Developed from the popular fully observable single-agent robot MuJoCo, MAMuJoCo is a collaborative multi-agent robot control platform with continuous action space and

observation space. In addition, since each agent only controls a part of the robot body, and the agent can only observe other agents adjacent to it, partial observability can be controlled in MAMuJoCo, and the higher speed can be reached only when all parts fully cooperate.

Following [3], we train 5 times more samples of the on-policy algorithm than the off-policy algorithm, because the sample efficiency of the off-policy algorithm is better than that of the on-policy algorithm. Specifically, in MAMuJoCo, on-policy algorithms such as MAPPG and MAPPO are trained with 10 million time steps, while MADDPG only has 2 million time steps. However, for convenience, the time step of the off-policy algorithm is extended by 5 times when drawing the figure, then it can be unified with the on-policy algorithm.

In addition, since MADDPG network parameters are much larger than the MAPPG and MAPPO algorithms[1], the local policy parameter sharing technology is not utilized for MAPPG and MAPPO, which greatly reduces the performance. Empirically, it is found that the MAPPO algorithm with completely heterogeneous policies has a similar performance to the HAPPO algorithm, thus we do not compare it with the HAPPO algorithm specifically.

It can be seen form Table II that the MAPPG algorithm exceeds the two benchmarks in a final performance in five tasks, while the MADDPG algorithm performs the worst. As shown in Fig. 4, the intermediate performance of the MAPPG algorithm in all three tasks also exceeds the two benchmarks. Especially in Fig. 4(b), the performance of MAPPO algorithm decreases significantly in the end, while MAPPG has certain robustness.

In summary, both qualitative and quantitative experimental results show that the proposed MAPPG algorithm outperforms the benchmark algorithm both in MPE and MAMuJoCo.

VI. CONCLUSIONS

This paper presented a multi-agent cooperative algorithm called MAPPG, conforming to the CTDE framework. Compared with the existing CTDE methods, the main highlight is that the local policy network is directly trained with environment states in the auxiliary phase, and the policy network is updated in a manner consistent with the general MD. Through a series of experiments on MPE and MAMuJoCo, the experimental results show that our proposed method is superior than the MAPPO and MADDPG benchmarks. Besides changing the structure of the local policy network, the introduction of the auxiliary phase does not involve the training of the policy phase, which can be naturally integrated into any CTDE framework. In future work, we aim to implement our frame-work in real-world robots, such as robot running competitions.

[1]The policy network of MAPPG algorithm is an MLP with 2 hidden layers of 64 units and ReLU non-linearities, while the actor network of MADDPG is an MLP with 2 hidden layers of 400 units.

REFERENCES

[1] S. Sarkar, V. Gundecha, A. Shmakov, S. Ghorbanpour, A. R. Babu, P. Faraboschi, M. Cocho, A. Pichard, and J. Fievez, "Multi-agent reinforcement learning controller to maximize energy efficiency for multi-generator industrial wave energy converter," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 11, 2022, pp. 12 135–12 144.

[2] M. Hüttenrauch, A. Šošić, and G. Neumann, "Guided deep reinforcement learning for swarm systems," *arXiv preprint arXiv:1709.06011*, 2017.

[3] G. Papoudakis, F. Christianos, L. Schäfer, and S. V. Albrecht, "Benchmarking multi-agent deep reinforcement learning algorithms in cooperative tasks," in *NeurIPS Datasets and Benchmarks*, 2020.

[4] C. S. de Witt, T. Gupta, D. Makoviichuk, V. Makoviychuk, P. H. Torr, M. Sun, and S. Whiteson, "Is independent learning all you need in the starcraft multi-agent challenge?" *arXiv preprint arXiv:2011.09533*, 2020.

[5] L. Kraemer and B. Banerjee, "Multi-agent reinforcement learning as a rehearsal for decentralized planning," *Neurocomputing*, vol. 190, pp. 82–94, 2016.

[6] C. Yu, A. Velu, E. Vinitsky, Y. Wang, A. Bayen, and Y. Wu, "The surprising effectiveness of ppo in cooperative, multi-agent games," *arXiv preprint arXiv:2103.01955*, 2021.

[7] T. Rashid, M. Samvelyan, C. S. De Witt, G. Farquhar, J. Foerster, and S. Whiteson, "Monotonic value function factorisation for deep multi-agent reinforcement learning," *The Journal of Machine Learning Research*, vol. 21, no. 1, pp. 7234–7284, 2020.

[8] K. W. Cobbe, J. Hilton, O. Klimov, and J. Schulman, "Phasic policy gradient," in *International Conference on Machine Learning*. PMLR, 2021, pp. 2020–2027.

[9] P. Sunehag, G. Lever, A. Gruslys, W. M. Czarnecki, V. Zambaldi, M. Jaderberg, M. Lanctot, N. Sonnerat, J. Z. Leibo, K. Tuyls *et al.*, "Value-decomposition networks for cooperative multi-agent learning," *arXiv preprint arXiv:1706.05296*, 2017.

[10] B. Peng, T. Rashid, C. Schroeder de Witt, P.-A. Kamienny, P. Torr, W. Böhmer, and S. Whiteson, "Facmac: Factored multi-agent centralised policy gradients," *Advances in Neural Information Processing Systems*, vol. 34, pp. 12 208–12 221, 2021.

[11] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *Advances in neural information processing systems*, vol. 30, 2017.

[12] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. M. O. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *CoRR*, vol. abs/1509.02971, 2015.

[13] E. Wei, D. Wicke, D. Freelan, and S. Luke, "Multiagent soft q-learning," *arXiv preprint arXiv:1804.09817*, 2018.

[14] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *ArXiv*, vol. abs/1707.06347, 2017.

[15] J. G. Kuba, R. Chen, M. Wen, Y. Wen, F. Sun, J. Wang, and Y. Yang, "Trust region policy optimisation in multi-agent reinforcement learning," *arXiv preprint arXiv:2109.11251*, 2021.

[16] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *Neural Information Processing Systems (NIPS)*, 2017.

[17] C. S. de Witt, B. Peng, P.-A. Kamienny, P. Torr, W. Böhmer, and S. Whiteson, "Deep multi-agent reinforcement learning for decentralized continuous cooperative control," *arXiv preprint arXiv:2003.06709*, vol. 19, 2020.

[18] T. Wang, T. Gupta, A. Mahajan, B. Peng, S. Whiteson, and C. Zhang, "Rode: Learning roles to decompose multi-agent tasks," *arXiv preprint arXiv:2010.01523*, 2020.

[19] F. A. Oliehoek and C. Amato, *A concise introduction to decentralized POMDPs*. Springer, 2016.

[20] M. Tomar, L. Shani, Y. Efroni, and M. Ghavamzadeh, "Mirror descent policy optimization," *arXiv preprint arXiv:2005.09814*, 2020.

[21] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," *arXiv preprint arXiv:1506.02438*, 2015.