2021 27th International Conference on Mechatronics and Machine Vision in Practice (M2VIP) | 978-1-6654-3153-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/M2VIP49856.2021.9665049

Learning Intra-group Cooperation in Multi-agent Systems

Weiwei Liu[1], Shanqi Liu[1], Jian Yang*[2] and Yong Liu*[1,3]

***Abstract*— Reinforcement learning is one of the algorithms used in multi-agent systems to promote agent cooperation. However, most current multi-agent reinforcement learning algorithms improve the communication capabilities of agents for cooperation, but the overall communication is costly and even harmful due to bandwidth limitations. In addition, decentralized execution cannot generate joint actions, which is not conducive to cooperation. Therefore, we proposed the Hierarchical Group Cooperation Network (HGCN). Advanced strategy, Group Network (GroNet), learns to group all agents based on their state rather than their location. The Low-level strategy, Group Cooperation Network (GCoNet), is a method of centralized training and centralized execution within a group, which effectively promotes agent collaboration. Finally, we validated our method in various experiments.**

I. INTRODUCTION

With the rapid development of reinforcement learning [1] in recent years, we have been able to see many algorithms that surpass human performance. For example: in the field of games, Alibaba proposed a Bi-directional recurrent neural network (RNN) [2] to learn the algorithm of the communication protocol between agents, Bidirectionally-Coordinated Net (BicNet) [3], which defeated the built-in artificial intelligence in StarCraft [4], [5]; AlphaGo [6] and AlphaZero [7] have gained worldwide attention in the field of Go; Tencent proposed the AI [8] in the King's Glory even defeated the top players. However, when a multi-intelligent system is in an environment known as a decentralized partially observable Markov decision progress (DEC-POMDP) [9], the agent only have access to local observations but cannot learn information about the observations, actions, and rewards of other agents, which can cause the algorithm to be unstable.

Scholars have proposed many methods to solve this problem. Multi-Agent Deep Deterministic Policy Gradient (MADDPG) [10] uses a centralized evaluation network to evaluate the state-actions of all agents jointly, but in order for all agents to make distributed decisions, each agent has its action network [11]. MADDPG solves the limitation of using experience playback technology and the convergence of common algorithms [12], [13] caused by a non-stationary Markov environment. In addition, MADDPG substantially alleviates the communication difficulties between agents and the credit assignment [14] of team rewards. However, the algorithm only takes the agent's observation as input in the decision-making stage, and it is easy to fall into a local optimum. Foerster [15] first proposed an explicit communication protocol between learning agents in 2016, but it assumes that the communication channel is discrete, so continuous information cannot be exchanged. The Reinforced Inter-Agent Learning (RIAL) [15] algorithm can share parameters between agents and explicitly transmit learnable information between agents to increase the agent's perception of the environment, but it lacks the ability to feedback communication behavior. Moreover, this algorithm is used in discrete environments.

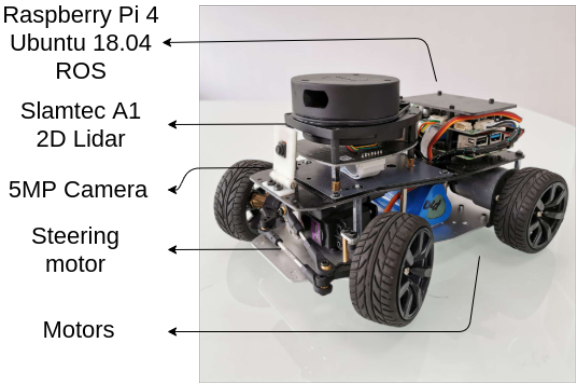


Fig. 1: Schematic diagram of the experimental car.

In order to promote cooperation between agents, We propose a novel algorithm for centralized training and execution within a group. We called hierarchical group collaboration network (HGCN). The algorithm constructs two reinforcement learning architectures, namely hierarchical reinforcement learning. The high-level algorithm divides all agents into groups. The divided group is regarded as one agent——the low-level algorithm inputs the local observations and actions of all agents in the group during training and execution, the Group Centralized Training and Centralized Execution (GCTCE). Finally, we introduce the VDN algorithm for value decomposition operation to solve the agent reputation assignment problem. The main contributions are as follows:

- To divide the group based on the information relationship between the agents rather than the position relationship. We use all agents' observations in the upper algorithm of the two-layer reinforcement learning to learn the agents' confidential information and then group them.
- We propose HGCN. It adopts centralized training and

This work was supported by the National Natural Science Foundation of China under Grant 62088101.

[1] the Advanced Perception on Robotics and Intelligent Learning Lab, College of Control Science and Enginneering, Zhejiang University, Hangzhou, China. [3] Huzhou Institute of Zhejiang University, Huzhou, China. [2] China Research and Development Academy of Machinery Equipment, Beijing, China.

*Corresponding authors: Jian Yang and Yong Liu, e-mail: buaayangjian@126.com and yongliu@iipc.zju.edu.cn.

execution in the group, so its communication difficulty is significantly reduced, effectively alleviating algorithm scalability caused by the increase in the number of agents.

- Finally, we verified the effectiveness of the algorithm in simulation and real-world scenarios.

II. RELATED WORK

The team rewards are assigned to different agent rewards according to the degree of contribution of each agent to the team, and then the objective function of each agent is optimized, that is, the credit assignment problem. Based on this motivation, Value-Decomposition Networks (VDN) [16] simply decomposes the team value function into the sum of the value functions of N agents and uses it as the evaluation standard of each action function to optimize the action function. The QMIX [17] algorithm is the follow-up work of the VDN algorithm. Its starting point is that the VDN only performs simple summation when doing joint Q-value decomposition. This approach will let the learned local Q function expression ability be limited, and there is no way to capture more complex relationships between agents. Therefore, QMIX generalizes the summation constraint to a larger family of monotonic functions. The above constraints are necessary and insufficient conditions for this family of monotonic functions.

Other scholars believe that the way to promote agent cooperation is to strengthen the communication between agents. Attentional Communication (ATOC) [18] has the option to communicate with other agents, which solves the communication problem of agents with collaborative value. ATOC proposes to use the attention mechanism to select agent communication, which alleviates the long delay and high computational complexity caused by a large amount of bandwidth required to receive a large amount of information. This is the first time that attention communication has been successfully applied to MARL. The difference between Multi-Actor-Attention-Critic (MAAC) [19] and ATOC is that it selectively accepts other processed information directly in the evaluation network, which is an end-to-end communication bandwidth reduction solution. Unlike MADDPG, MAAC's action is sampled from the current strategy function, not from the replay buffer, because sampling from the replay buffer will lead to overgeneralization, making it impossible for agents to coordinate the current strategy effectively.

Multi-Agent Proximal Policy Optimization (MAPPO) [20] uses a central value function method to consider global information, and complete communication between agents is a method within the scope of the Centralized Training and Decentralized Execution (CTDE) framework. A global value function is used to make each individual Proximal Policy Optimization (PPO) [21] agent cooperate with each other. It has a predecessor Independent Proximal Policy Optimization (IPPO) [22], which is an entirely decentralized PPO algorithm, similar to the Independent Q-learning (IQL) [23] algorithm.

III. METHODS

This work's starting point, in large-scale multi-agent systems, is challenging to give each agent a separate reward function. Generally, all agents share a joint reward function, which also brings about the agent's credit distribution problem– the lazy and hardworking agent's rewards are the same. Peter Sunehag proposed the VDN algorithm, which can only be used in a small number of agents in a cooperative and discrete environment. On this basis, we propose a novel algorithm: HGCN, and its structure is shown in Figure 2.

A. Hierarchical group collaboration network

Suppose we have N agents, which can be divided into K groups. As shown in Figure 2, the high-level grouping algorithm divides groups based on all agents' observations. As the lower layer group collaboration algorithm's strategy is constantly changing, the grouping algorithm adopts the actor-critic [24]. Set the actor and critic network (μ_g, Q_g) parameters in the grouping algorithm as θ_g^μ and θ_g^Q, the target network $(\overline{\mu}_g, \overline{Q}_g)$ parameters are $\overline{\theta}_g^\mu$ and $\overline{\theta}_g^Q$. The training tuple is $(O, A, R, O^{'})$. Where $O = (o_1, \cdots, o_n)$ $O^{'} = (o_1^{'}, \cdots, o_n^{'})$, they respectively represent the joint state of all agents at the current and next moments. The action A dimension is consistent with the number of agents. At this time, each agent has a score, which is sorted and grouped by score. The reward function R—the global reward—is the same as the underlying group collaboration algorithm. Updating the state-action value function Q_g as:

$$\begin{aligned} \mathcal{L}\left(\theta_g^Q\right) &= \mathbb{E}_{(O,A,R,O')}\left[\left(Q_g(O,A) - y\right)^2\right], \\ y &= r + \gamma \overline{Q}_g(O^{'}, A^{'})|_{A'=\overline{\mu}_g(O');\overline{\theta}_g^Q}. \end{aligned} \tag{1}$$

The group policy gradient can be written as:

$$\nabla_{\theta_g^\mu} J\left(\theta_g^\mu\right) = \mathbb{E}_{(O,A)}\left[\nabla_{\theta_g^\mu} \mu_g(A \mid O) \nabla_A Q_g \mid_{A=\mu_g(O)}\right]. \tag{2}$$

Because it takes some time for agent cooperation to show effect, the grouping network's updated range cannot be too extensive. In this way, in a relatively short period, each group's members change very little. We add a penalty item to the grouping algorithm's policy update, which is the KL divergence between the current and the previous policy distribution.

$$\begin{aligned} \nabla_{\theta_g^\mu} J\left(\theta_g^\mu\right) = \mathbb{E}_{(O,A)}&\left[\nabla_{\theta_g^\mu} \mu_g(A \mid O) \nabla_A Q_g \mid_{A=\mu_g(O)}\right] \\ &- \alpha D_{KL}\left[\pi_{\theta_{old}^\mu}(\cdot \mid s) \parallel \pi_{\theta_{new}^\mu}(\cdot \mid s)\right], \end{aligned} \tag{3}$$

where α is the penalty coefficient.

Use VDN for value decomposition. The DDPG algorithm is used for training.

The training state-action value function is:

$$\begin{aligned} \mathcal{L} &= \mathbb{E}_{(O_g, A_g, R, O_g')\backsim\mathbb{R}}\left[\left(Q_{tot}(O_{tot}, A_{tot}) - y\right)^2\right], \\ y &= r + \gamma \overline{Q}_{tot}(O_{tot}^{'}, A_{tot}^{'}), \end{aligned} \tag{4}$$

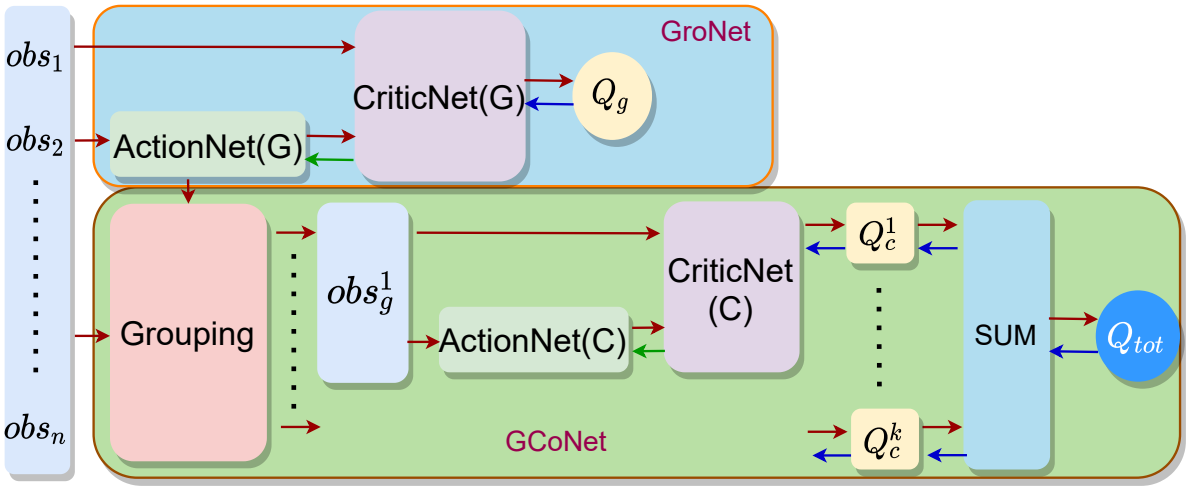


Fig. 2: Hierarchical group collaboration network framework diagram. HGCN can be divided into two networks according to the upper and lower structure (GroNet and GCoNet), GroNet represents the grouping network whose input is the state of all agents, and GCoNet represents the group cooperation network.

Algorithm 1 HGCN+KL

Randomly initialize separately GroNet and GCoNet's actor and critic network $\mu_g\left(O|\,\theta_g^\mu\right)$, $Q_g\left(O,A|\,\theta_g^Q\right)$ and $\mu_c\left(O_g|\,\theta_c^\mu\right)$, $Q_c\left(O_g,A_g|\,\theta_g^c\right)$ with weights θ_g^μ, θ_g^Q and θ_c^μ, θ_c^Q.
Initialize separately GroNet and GCoNet's target actor and critic network $\overline{\mu}_g$, $\overline{Q}_g$ and $\overline{\mu}_c$, $\overline{Q}_c$ with weights $\overline{\theta}_g^\mu \leftarrow \theta_g^\mu$, $\overline{\theta}_g^Q \leftarrow \theta_g^Q$ and $\overline{\theta}_c^\mu \leftarrow \theta_c^\mu$, $\overline{\theta}_c^Q \leftarrow \theta_c^Q$.
Initialize replay buffer R.
for episode=1, M **do**
Initialize a random process $\mathcal{N}$ for action exploration.
Receive initial Observation state O.
for t=1, T **do**
Select action $a_g = \mu_g\left(O|\,\theta_g^\mu\right)$, Sort grouping and get $(O_g^1,\ldots,O_g^k)$; Select action $A_g = \mu_c\left(O_g|\,\theta_c^\mu\right) + \mathcal{N}_t$.
Execute action A_t and observe reward r_t and observe new state O.
Store transition (O_g, A_g, r_t, O_g')
Update critic networks by formula (1) and (4).
Update actor networks by formula (3) and (6).
Update the target networks:

$$\overline{\theta}_g \leftarrow \tau\theta_g + (1+\tau)\overline{\theta}_g$$
$$\overline{\theta}_c \leftarrow \tau\theta_c + (1+\tau)\overline{\theta}_c$$

end for
end for

$$Q_{tot}(O_{tot},A_{tot}) = Q_c^1(O_g^1,A_g^1) + \cdots + Q_c^k(O_g^k,A_g^k),$$
$$\overline{Q}_{tot}(O_{tot}',A_{tot}') = \overline{Q}_c^1(O_g'^1,A_g'^1) + \cdots + \overline{Q}_c^k(O_g'^k,A_g'^k), \quad (5)$$

The network gradient is:

$$\nabla_{\theta_c^\mu} J\left(\theta_g^\mu\right) = \mathbb{E}_{(O_g,A_g)}\left[\nabla_{\theta_c^\mu}\mu_c\left(A_c \mid O_c\right)\nabla_{A_c}Q_c\mid_{A_c=\mu_c(O_c)}\right]. \quad (6)$$

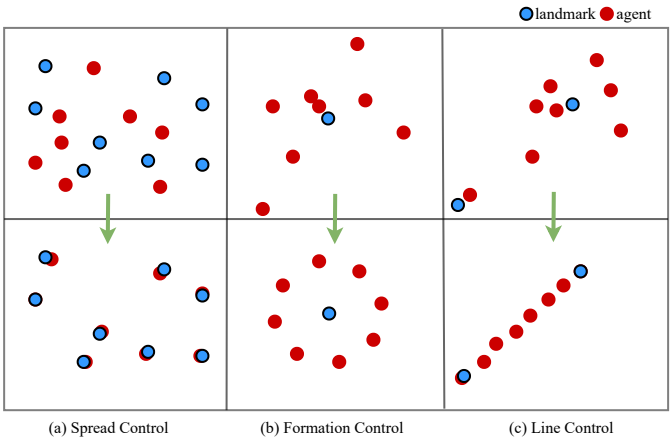


Fig. 3: Illustration of multi-agents learning to cooperate in three scenarios.

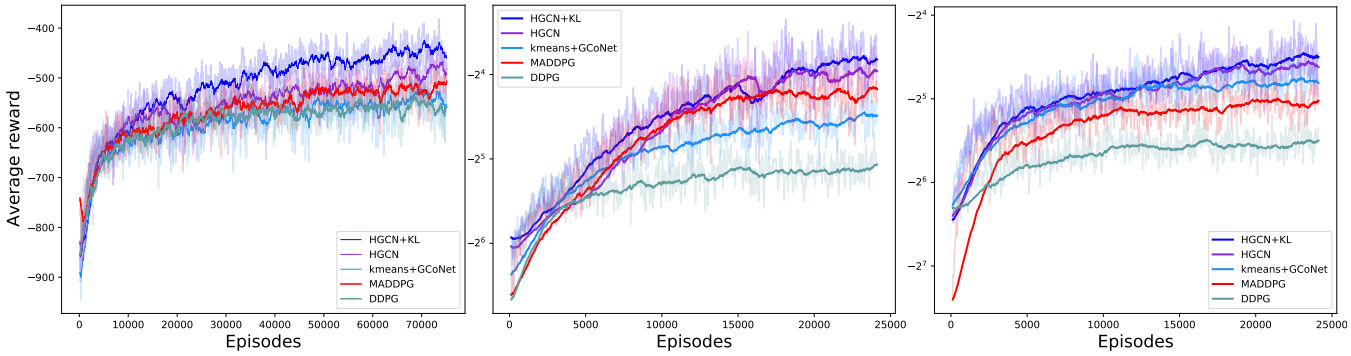
Soft update: $\theta = \tau\theta + (1-\tau)\theta$.

IV. EXPERIMENT

Lowe et al. proposed an open source multi-agent environment-Multi-Agent Particle Environment [25]. We use three standard tasks [26] [27] to evaluate HGCN. These tasks are shown in Figure 3. The baseline algorithm for comparison uses the well-known MADDPG and DDPG. All agents share the same reward function.

A. Experimental results and analysis

As shown in Figure 3, due to the time continuity of the cooperation between agents, we added the penalty of KL divergence to the loss function of GroNet's action network update so that the grouping algorithm is limited to update, which we use HGCN+KL to express. Pure HGCN is an ablation experiment. In order to test the effect of general



(a) Spread control (b) Formation control (c) Line control

Fig. 4: Illustration of multi-agents learning to cooperate in three scenarios.

TABLE I: Experiment site and trolley parameter table.

Field length	Maximum linear speed	Maximum acceleration	Maximum angular acceleration	Wheelbase
$4m$	$0.6m/s$	$0.5m/s^2$	$0.5rad/s^2$	$0.145m$
Field width	Minimum line speed	Minimum turning radius	Car length	Car width
$2.3m$	$-0.2m/s$	$0.375m$	$0.22m$	$0.185m$

algorithm grouping according to agents' location, we use the K-means algorithm to group agents and combine it with GCoNet to verify the necessity of our proposed GroNet.

Spread Control: In this task, the agents choose their respective landmarks as their destinations after communicating with each other to avoid conflicts. Since agents collide with each other, they will lose points, so try to avoid collisions during the movement. Figure 4(a) shows the average score of the agent under each algorithm. HGCN+KL achieved the best score and performed better than HGCN, which indicates that it takes a certain amount of time for the results of cooperation between agents to appear. The continuous grouping of agents is not conducive to their cooperation. kmeans+GCoNet is lower than HGCN and even lower than MADDPG, which shows that grouping based on agents' relative position is not conducive to cooperation between them. The worst effect of DDPG shows that in a multi-agent system with a strong coupling and dynamic environment, the agent does not consider other agents but only relies on its state to make decisions, and it is not easy to form practical cooperation with other agents.

Formation Control: As shown in Figure 4(b), since we canceled the measure of losing points due to collision between agents, the rate of increase in the reward curve for each round changes little. Moreover, the location of the landmarks in this task is more regular than the first task, and the algorithm fluctuations are negligible.

Line Control: As shown in Figure 5, although the overall score trend remains unchanged, the kmeans+GCoNet algorithm score in this task is higher than that of MADDPG. Compared with the other two tasks, since the agent needs to line up in a straight line and the space between the two points is limited, the agent is prone to the collision at the beginning, but once the agent learns to avoid collisions, the rate of increase of the reward curve in Figure 4(c) will be extremely fast, even if the task is not completed.

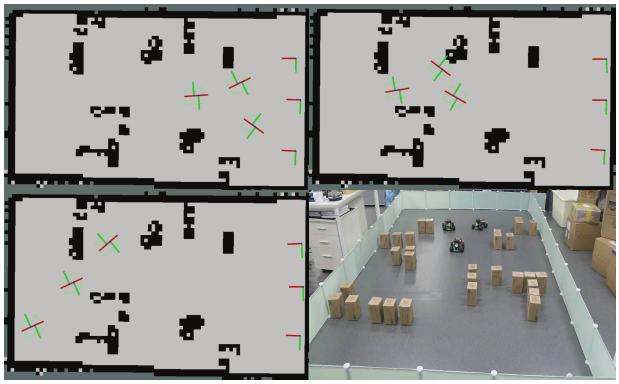


Fig. 5: Hardware experiment diagram, Lower left corner: the real car position; others: the radar positioning position of the agent in rviz.

B. Hardware experiment

We conducted experiments on the hardware platform. The experimental car is shown in Figure 1. The parameters of the trolley are shown in Table 1. The experimental process is shown in Figure 7. Due to the limitation of the experimental site, we chose three cars to conduct the formation experiment. This experimental result proves the transferability of our method from simulation to experiment.

V. CONCLUSIONS

In order to solve the problem that the reinforcement learning algorithm of global centralized training is challenging to extend to large-scale multi-agent systems and the decentralized execution is not conducive to cooperation between agents, this article proposes a method of group centralized training and centralized execution: HGCN. To group more effectively, rather than according to the relative position of the agent, we propose the grouping network GroNet, which

forms hierarchical reinforcement learning with GCoNet. Finally, we conducted experiments on three standard multi-agent robot tasks and achieved excellent performance.

REFERENCES

[1] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[2] Y. Wang, M. Long, J. Wang, Z. Gao, and P. S. Yu, "Predrnn: Recurrent neural networks for predictive learning using spatiotemporal lstms," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 879–888, 2017.

[3] P. Peng, Y. Wen, Y. Yang, Q. Yuan, Z. Tang, H. Long, and J. Wang, "Multiagent bidirectionally-coordinated nets: Emergence of human-level coordination in learning to play starcraft combat games," *arXiv preprint arXiv:1703.10069*, 2017.

[4] W. Wang, T. Yang, Y. Liu, J. Hao, X. Hao, Y. Hu, Y. Chen, C. Fan, and Y. Gao, "From few to more: Large-scale dynamic multiagent curriculum learning.," in *AAAI*, pp. 7293–7300, 2020.

[5] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, *et al.*, "Grandmaster level in starcraft ii using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.

[6] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[7] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, *et al.*, "A general reinforcement learning algorithm that masters chess, shogi, and go through self-play," *Science*, vol. 362, no. 6419, pp. 1140–1144, 2018.

[8] D. Ye, Z. Liu, M. Sun, B. Shi, P. Zhao, H. Wu, H. Yu, S. Yang, X. Wu, Q. Guo, *et al.*, "Mastering complex control in moba games with deep reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, pp. 6672–6679, 2020.

[9] S. Omidshafiei, A.-A. Agha-Mohammadi, C. Amato, S.-Y. Liu, J. P. How, and J. Vian, "Decentralized control of multi-robot partially observable markov decision processes using belief space macro-actions," *The International Journal of Robotics Research*, vol. 36, no. 2, pp. 231–258, 2017.

[10] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Advances in neural information processing systems*, pp. 6379–6390, 2017.

[11] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[12] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel, "Benchmarking deep reinforcement learning for continuous control," in *International conference on machine learning*, pp. 1329–1338, PMLR, 2016.

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

[14] R. S. Sutton, *Temporal credit assignment in reinforcement learning*. PhD thesis, University of Massachusetts Amherst, 1984.

[15] J. N. Foerster, Y. M. Assael, N. de Freitas, and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," in *NIPS*, 2016.

[16] 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 based on team reward," in *PROCEEDINGS OF THE 17TH INTERNATIONAL CONFERENCE ON AUTONOMOUS AGENTS AND MULTIAGENT SYSTEMS (AAMAS'18)*, vol. 3, pp. 2085–2087, ASSOC COMPUTING MACHINERY, 2018.

[17] T. Rashid, M. Samvelyan, C. Schroeder, G. Farquhar, J. Foerster, and S. Whiteson, "Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning," in *International Conference on Machine Learning*, pp. 4295–4304, PMLR, 2018.

[18] J. Jiang and Z. Lu, "Learning attentional communication for multi-agent cooperation," in *Advances in neural information processing systems*, pp. 7254–7264, 2018.

[19] S. Iqbal and F. Sha, "Actor-attention-critic for multi-agent reinforcement learning," in *International Conference on Machine Learning*, pp. 2961–2970, PMLR, 2019.

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

[21] Y. Wang, H. He, and X. Tan, "Truly proximal policy optimization," in *Uncertainty in Artificial Intelligence*, pp. 113–122, PMLR, 2020.

[22] 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.

[23] A. Tampuu, T. Matiisen, D. Kodelja, I. Kuzovkin, K. Korjus, J. Aru, J. Aru, and R. Vicente, "Multiagent cooperation and competition with deep reinforcement learning," *PloS one*, vol. 12, no. 4, p. e0172395, 2017.

[24] T. Degris, M. White, and R. S. Sutton, "Off-policy actor-critic," in *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pp. 179–186, 2012.

[25] I. Mordatch and P. Abbeel, "Emergence of grounded compositional language in multi-agent populations," in *Thirty-second AAAI conference on artificial intelligence*, 2018.

[26] M. Mesbahi and M. Egerstedt, *Graph theoretic methods in multiagent networks*, vol. 33. Princeton University Press, 2010.

[27] T. Balch and R. C. Arkin, "Behavior-based formation control for multirobot teams," *IEEE transactions on robotics and automation*, vol. 14, no. 6, pp. 926–939, 1998.