Proceeding of the IEEE
International Conference on Robotics and Biomimetics
Dali, China, December 2019

The Effect of Different Types of Internal Rewards in Distributed Multi-Agent Deep Reinforcement Learning[*]

Hongda Zhang
State Key Laboratory of Robotics
Shenyang Institute of Automation
Chinese Academy of Sciences
Shenyang, Liaoning 110016, China
Institutes for Robotics and Intelligent Manufacturing
Chinese Academy of Sciences
Shenyang, Liaoning 110016, China
University of Chinese Academy of Sciences
Beijing 100049, China
zhanghongda@sia.cn

Decai Li and Yuqing He
State Key Laboratory of Robotics
Shenyang Institute of Automation
Chinese Academy of Science
Shenyang, Liaoning 110016, China
Institutes for Robotics and Intelligent Manufacturing
Chinese Academy of Sciences
Shenyang, Liaoning 110016, China
{lidecai & heyuqing}@sia.cn

Abstract - **Distributed multiagent reinforcement learning in the same environment is prohibitively hard, due to the difficulty of assigning credit for the individual actions of the agent, especially when the agent is a member of a team. Meanwhile, the sparse delayed reward about the team from the environment such as winning makes the learning progress more challenging. To solve the credit assignment and sparse delayed reward problems which are common in multiagent reinforcement learning, researchers usually construct or learn an internal reward signal that acts as a proxy for winning and provides denser rewards for individual agent. To improve the learning effect of a typical multiagent learning task, we conducted three types of internal rewards for multiagent team members and evaluated the effect of these rewards. The results show that not all internal reward can improve the learning effect of multiagent reinforcement learning, it seems that when the learning task is not very complex and the time of finishing the task for the team is not very long, the sparse reward such as winning have the best learning effect, and the learning effect of the other two forms of reward is not as good as that of the simple sparse reward. To some extent, our research results can provide a reference for the design of reward function in the application of distributed multi-agent reinforcement learning.**

Index Terms - Internal reward, credit assignment, multiagent, deep reinforcement learning, distributed.

I. Introduction

In the field of mobile robots, the advantages of multi-robot systems over single-robot systems have been widely recognized by researchers [1], and multi-robot collaboration has become one of the main development trends of mobile robots. Among them, multi-robot collaborative decision-making is the necessary foundation to achieve multi-robot coordination. The difficulties and challenges of multi-robot collaborative decision-making include: high dimensionality of solving problems, strong dynamics and time-varying in adversarial, incomplete information, time/space joint reasoning and distributed decision making.

The multi-agent deep reinforcement learning method generated by deep learning combined with multi-agent reinforcement learning can make high-dimensional space and space-time inference decision making. Numerous research institutes have conducted a large number of researches, such as Bicnet, COMA, MADDPG [2], A3C, hierarchical reinforcement learning, hierarchical time reinforcement learning and so on. These multi-agent deep reinforcement learning methods have been successfully applied in multi-agent adversarial games such as StarCraft II, DOTA 2, Arena of Valor, and football video games.

However, the multi-agent deep reinforcement learning method still has problems in the multi-robot collaborative decision-making application in the actual adversarial environment. For example, the robot has sensor error and individual fail, which leads to strong uncertainty of observation information. Communication constraints such as communication bandwidth, distance limitation and information unmeasurable between robots lead to information incompleteness. Wide-ranging environment, distributed tasks, mobile computing resources and other constraints make it only have one solution, distributed decision making.

The distributed multi-agent reinforcement learning method is an effective method to solve the multi-robot distributed decision problem under the condition of incomplete information. The great challenge of this type of approach is that distributed multi-agent reinforcement learning in the same environment is very difficult. Because it is difficult to assign rewards to the individual behavior of the agent, especially when the agent is a team member. At the same time, the sparse delay signals such as “victory” that the team gets from the environment make the learning process more challenging. In order to solve the problem of credit allocation and sparse delay compensation commonly found in multi-agent reinforcement learning, researchers usually construct or learn an internal reward signal as a winning proxy to provide more intensive rewards for agents. In order to improve the learning effect of typical multi-agent learning

[*] This work is supported by NSFC 91748130, NSFC U1608253 and Youth Innovation Promotion Association, CAS.

tasks, we constructed three types of internal rewards for multi-agent team members and evaluated the effects of these rewards. To a certain extent, our research results can provide reference for the design of reward function in the process of distributed multi-agent reinforcement learning application.

II. Related Work

In order to solve the problem of credit allocation and sparse delay compensation commonly found in distributed multi-agent reinforcement learning, researchers usually construct or learn an internal reward signal to provide more intensive rewards for individual agents, which facilitates the rapid convergence and improvement of the strategy learning process and strategy learning effect.

Some researchers have verified that deep reinforcement learning is effective in dealing with adversarial problems and delay rewards. Wender [3] applied variants of the Q-learning and Sarsa algorithms, using qualification traces to offset the problem of delayed rewards. Kempka [4] verified the feasibility of visual reinforcement learning in a three-dimensional first-person perspective environment, VizDoom. In a basic motion and shooting task and a more complex labyrinth navigation scenario, a deep convolutional neural network with Q learning and experience playback can be used to train autonomous game programs that exhibit human behavior. Usunier [5] proposed a method for deep neural network controllers to deal with micro-management scenarios from the original state features given by the game engine, and solved the short-term low-level control problems of military members in combat of video game. The algorithm uses deterministic strategy to collect traces of learning, which is more effective than "beast-like exploration". But these methods become difficult to learn in multi-agent distributed collaborative strategy learning.

It is a direct solution to build a suitable communication network for multiple agents or to develop a training method that can solve multi-agent collaborative strategies. Peng [6] introduced a multi-agent two-direction coordination network, BiCNet, in order to maintain a scalable and effective communication protocol while coordinating multiple team operations to defeat enemy missions in video game. The network contains a vectorized extended actor-critic formula that can handle battles of any number of different types of agents on both sides of the battle. In the absence of any data such as human demonstration or tagging, BiCNet can learn advanced coordination strategies commonly used by experienced gamers. Foerste [7] proposed a multi-agent Actor-Critic method with a counter-factual multi-agent (COMA) strategy gradient. COMA uses a centralized critic to estimate the Q function and a distributed actor to optimize the agent's strategy. To address the challenge of multi-agent credit allocation, it uses a counterfactual baseline to marginalize the behavior of an agent while keeping other agents' behaviors fixed. Vinyals [8] introduced the initial baseline results for a typical deep reinforcement learning agent for the video game StarCraft II. In mini-games, these agents can learn to reach the level of the game equivalent to novice players. However, these agents are unable to make significant progress in the training of complete games.

Distributed multi-agent reinforcement learning combined with constructing internal rewards and other methods to solve delayed rewards brings good results for multi-agent distributed collaborative strategy learning. Lanctot [9] introduced a new metric, joint policy association, to solve the problem of using the Independent Reinforcement Learning (InRL) strategy in multi-agent reinforcement learning (MARL) to fit other agent strategies during training. To quantify this effect, at the same time, a general MARL algorithm is proposed, which is based on the approximate optimal response of the strategy blending generated by deep reinforcement learning and the empirical game analysis to calculate the meta-strategy of strategy selection. Jaderberg [10] used a two-layer optimization approach in the first-view multiplayer game. A group of independent reinforcement learning agents play against opponents in a randomly generated environment in the form of teams through thousands of parallel games. Each of the individuals in the group learns its own internal rewards to supplement the winning sparse delay rewards and uses novel time-hierarchical representations to select actions so that the agent can reason on multiple time scales. Lowe [2] proposed a multi-agent distributed deep reinforcement learning method MADDPG. They are used for solving the difficulty of multi-agent environment faced by traditional algorithms, Q-learning, will be challenged by environmental instability, and the policy gradient method complexity grows exponentially when the number of agents increases. The variant of the actor-critic method can successfully learn the multi-agent collaborative policy while considering the action policy of other agents. They demonstrate the advantages of the proposed approach in an environment of cooperation and competition compared to existing methods, in which the agent community can discover cooperative strategies at various physical and information levels.

III. Background

We conducted our research based on the classical method MADDPG in distributed multi-agent reinforcement learning. Due to the length of the article, for more information about MADDPG and multi-agent reinforcement learning, please refer to [2].

IV. Methods

We evaluate the effectiveness of our rewards in a multi-agent collaborative and adversarial environment. In order to better understand the meaning of the form of reward form, we first introduce the environment used in our strategy training and evaluation.

A. *Experiments Environment*

Predator-prey [2][11]. In this variant of the classic predator-prey game, three slower cooperative agents (predator, red color) must catch up with faster opponents (prey, green color) in a randomly generated environment, while two large landmarks (black color) hinder the road. In general, whenever a cooperative predator collides with a hunt for prey, the captured agent will be rewarded, and the preyed agent will be punished. Therefore, the purpose of the predator is to catch the prey more quickly. In addition to its speed and position, the predator can also observe the relative position and velocity of the prey agent and the location of the landmark. In addition to its own speed and position, the prey can observe the relative position of the predator agent and the location of the landmark.

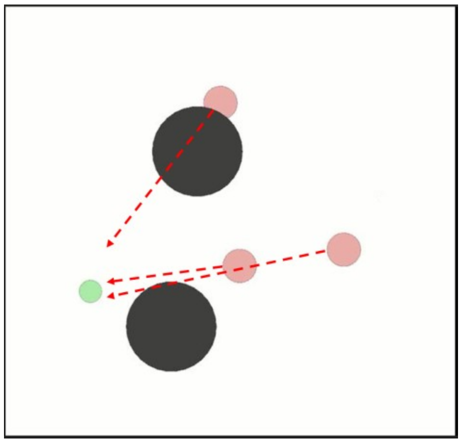

Fig. 1 The environment of predator-prey.

The choice of action strategy of the predator and the prey is (*noop, left, right, down, up*). In each direction, the maximum acceleration of the prey is 4.0, and the maximum acceleration of the predator is 3.0. The maximum speed of the prey is 1.3, and the maximum speed of the predator is 1.0.

The coordinate value of the adversarial environment area is between [-1, 1]. The environmental time step is 0.1s/step, the total step of each countermeasure environment is 25 steps, and the total time of each adversarial environment is 2.5s.

In the setting of adversarial environment, when any entity (predator, prey, obstacle) collides, it will generate the opposite force caused by collision. The reverse force makes each movable entity (predator, prey) produce acceleration in the direction of the force, so as to separate the movable entity for a certain distance. When the reverse force disappears, the mobile entity continues to take actions according to the actions given by the policy network. Here, we set that all collisions with landmarks will not be punished, and the collision between mobile entities is a sign of successful capture, which will generate corresponding reward and punishment information for the predator and the prey.

B. Three Different Forms of Rewards

Based on the MADDPG method and above environment, we designed three forms of rewards, called simple rewards, speed type rewards, and acceleration type rewards.

Before we start introducing the three types of rewards, because in the adversarial environment, we want the prey to run inside of our constructed environment as much as possible. To this end, we construct a boundary penalty for the prey, in the form as follows

$$\mathrm{b}(x)=\begin{cases} 0, \quad x<0.9; \\ 10*(x-0.9), \quad 0.9\le x<1; \\ \min(e^{2x-2},10), \quad 1\le x. \end{cases} \tag{1}$$

Where *b(x)* refers to the absolute value of the coordinates in the environment in which the prey is located, ie the absolute value of x and y in *(x, y)*. This equation shows that when the prey is close enough to the edge of the screen, that is, the edge of the environment, it will get a certain penalty. As the distance of prey to the edges get closer and outside of the environment, the penalty is getting more. Therefore, the penalty can prevent the prey from running out of the environment, so that we can train and observe the effect of the strategy in the environment. This part of the penalty set for the prey exists as part of the reward in the three forms of reward for the prey. For the predator, the purpose is to hunt down the prey, and the construction strategy can often show a certain pursuit effect, so we do not set a penalty for getting close to the edge or going out of the environment.

1) Simple Reward: Simple rewards are rewards with the final outcome information of the confrontation. In the adversarial environment, for the prey, each time it is arrested by any of the predator, the reward is -10. Therefore, the strategy for preys to learn is to avoid being punished for being arrested. Its total reward is

$$r_a=\begin{cases} -b(x) \\ -10 \end{cases} \tag{2}$$

For the predator, any predator who arrests the prey will receive a +10 bonus for all the predators. Therefore, the strategy of the team of predators is to capture the prey as much as possible.

2) Speed Type Reward: Speed-type rewards use the motion information of each entity, that is, position change information. We call this type of reward form as speed-type reward. For preys, the reward component adds reward information of distance to all individuals in the team of the predator. The prey is distanced from all predators and multiplied by a factor as part of their current reward. Therefore, the reward for the prey is

$$r_a=\begin{cases} -b(x) \\ -10, \quad if \quad caught \\ +0.1*\sum_{i=1}^{3} d_{a\sim p_i} \quad i=1,2,3 \end{cases} \tag{3}$$

Where $d_{a\sim p_i}$ is the distance between the prey and the predator. Therefore, the reward form not only has the effect of a simple reward, but also guides the prey to increase the distance from the predator.

For the predator, the reward component adds reward information of distance to the prey. That is, the minimum distance between the predator and the prey is multiplied by a factor as part of the penalty in its current reward information. Therefore, the reward of the pursuer is

$$r_p = \begin{cases} +10, & if \quad captured \\ -0.1 * \min d_{a\sim p_i}, & i = 1,2,3 \end{cases} \tag{4}$$

Where $d_{a\sim p_i}$ is the distance between the predator and the prey. Therefore, this form of reward not only has the effect of a simple reward, but also guides the predator to keep the distance from the prey reduced.

3) Acceleration Type Reward: The acceleration type reward uses the information of the position change rate of each moving entity, that is, the speed change information, so we call this type of reward form an acceleration type reward. For prey, the reward component adds reward information for the sum of distance change with all individuals of the team of the predator in the continuous time step. The amount of change in the distance between the prey and all the predators is taken as part of their current reward. Therefore, the reward for the prey is

$$r_a = \begin{cases} -b(x) \\ -10, \quad if \quad caught \\ +\sum_{i=1}^{3}\left(d^t_{a\sim p_i} - d^{t-1}_{a\sim p_i}\right), \quad i = 1,2,3 \end{cases} \tag{5}$$

Where $d^t_{a\sim p_i}$ is the distance between the prey and the predator at time *t*. Therefore, the reward form not only has the effect of a simple reward, but also guides the prey to keep increasing the distance from the predator for continuous time.

For the predator, the reward component adds the reward information for the distance between the predator and prey in the adjacent time step. That is, the minimum distance change between the predator and the prey in two adjacent time steps as part of the penalty in predator's current reward information. Therefore, the reward of the predator is

$$r_p = \begin{cases} +10, & if \quad captured \\ -(\min d^t_{a\sim p_i} - \min d^{t-1}_{a\sim p_i}), & i = 1,2,3 \end{cases} \tag{6}$$

Where $d^t_{a\sim p_i}$ is the distance between the prey and the predator at time *t*. Therefore, this form of reward not only has the effect of a simple reward, but also guides the predator to keep reducing the distance from the escaper for continuous time.

V. Experiments

A. Training Parameters

We use the MADDPG algorithm to evaluate the different forms of rewards we propose in the environment described in IV A. Using the distributed actor-critic method, the policy network is a two-layer ReLU MLP with 64 units per layer. The value network is also a two-layer ReLU MLP with 64 units per layer. The parameters of the network are updated as described above. The parameters in the environment and training process are shown in Table 1. We set the maximum time step for each adversarial process as 25 time steps. In the predator-prey environment, up to 60,000 adversarial scenarios were trained, with a total of 1.5 million time steps per training session. The training data is recorded every 200 time steps for visual display of the training results. In the end, we draw out the prey individual, the predator average and the whole environment rewards obtained by the process vary with the progress of the 60,000 sessions and 1.5 million time steps training process.

In order to compare the training effects of three different forms of rewards in the training process, we respectively carried out 25,000 time steps of the adversarial environment with the already trained convergence network, that is, 1000 iterations of the adversarial process. At the same time, different reward forms are combined to further explore the relationship between training effects and changes in reward forms.

TABLE I
List of Training Parameters

Training parameters	Value	Core training parameters	Value
Max episode length	25	Discount factor	0.95
Number of episodes	60000	Learning rate (begin)	0.01
Save rate (steps)	200	Batch size	1024
Benchmark iters (steps)	25000	Number of units (per layer)	64

B. Training Effect Comparison of the Different Forms of Rewards

We evaluated the trained strategies of several different forms and their combined forms of rewards. That is, 25,000 time steps are performed for each policy network, ie 1000 iterations of the adversarial process. Perform 10 times of such assessments for each strategy network and take the average of their success in each of the adversarial environments. We record the results of the final evaluation of each strategy network as follows.

TABLE II
Average Number of Successes in the Strategy of Different Rewards in Each Round of confrontation

Type of reward	Both simple reward	Both speed type reward	Both accele-ration type reward:	Prey use simple reward and predator use speed type reward	Prey Use speed type reward and predator use simple reward
Average number of successes	1.46	1.08	1.03	1.15	1.45

We record the training process for different reward forms as follows. In each case of reward form, the prey and the average of the three predators and the whole adversarial environment of the 60,000 adversarial process, ie 1.5 million time steps real-time reward were recorded.

1) Both Simple Reward:

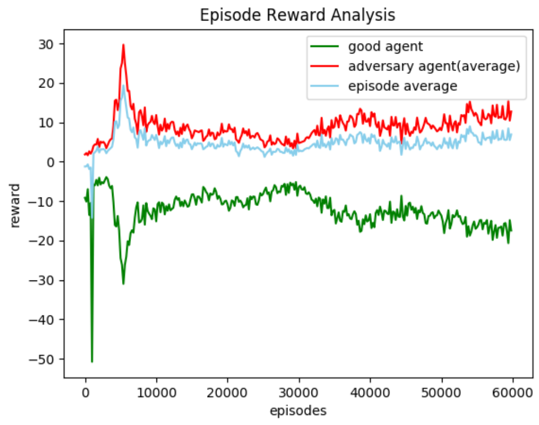

Fig. 2 The 60,000 adversarial process real-time reward of *Both Simple Reward*.

2) Both Speed Type Reward:

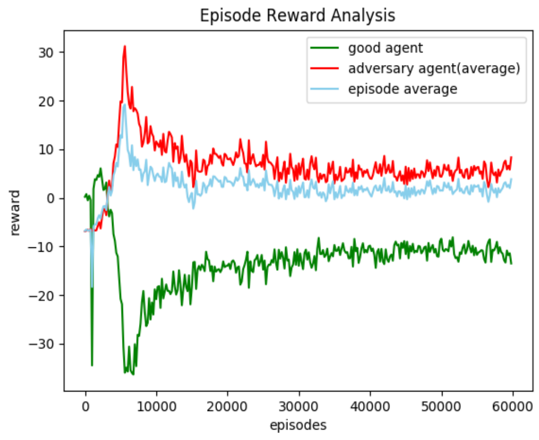

Fig. 3 The 60,000 adversarial process real-time reward of *Both Speed Type Reward*.

3) Both Acceleration Type Reward:

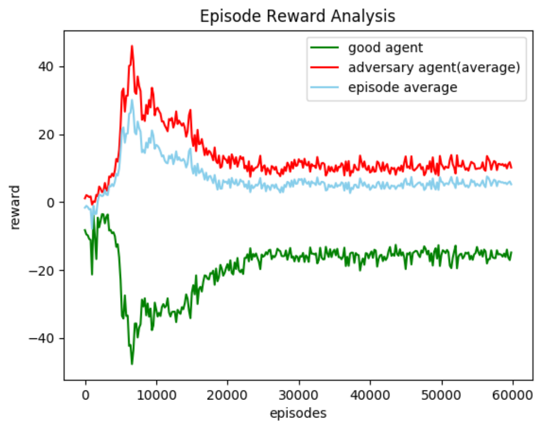

Fig. 4 The 60,000 adversarial process real-time reward of *Both Acceleration Type Reward*.

4) Prey Use Simple Reward and Predator Use Speed Type Reward:

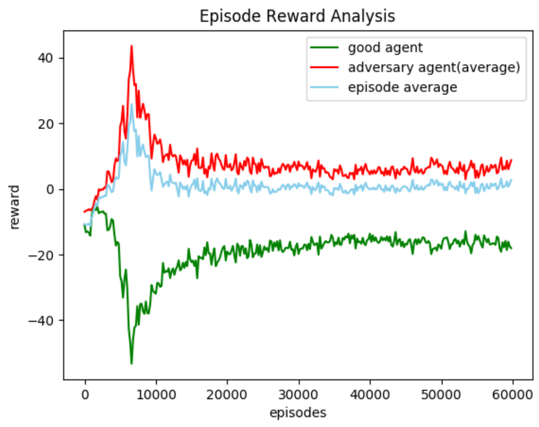

Fig. 5 The 60,000 adversarial process real-time reward of *Prey Use Simple Reward and Predator Use Speed Type Reward*.

5) Prey Use Speed Type Reward and Predator Use Simple Reward:

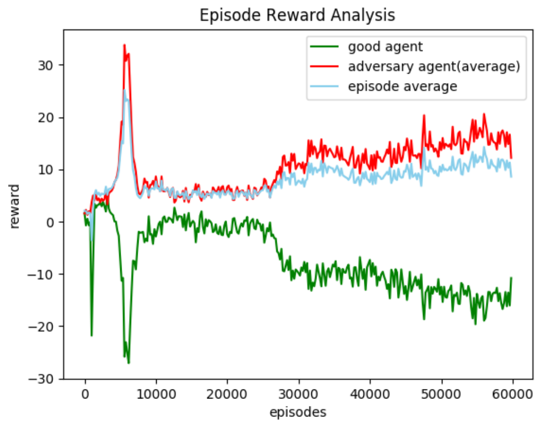

Fig. 6 The 60,000 adversarial process real-time reward of *Prey Use Speed Type Reward and Predator Use Simple Reward*.

D. Result Analysis

The results of Table 2 and Fig 2, 3, and 4 show that these more complex internal rewards such as *Speed Type Reward* and *Acceleration Type Reward* do not necessarily improve the effect of learning methods. Even the seemingly reasonable internal reward form does not improve the training effect of the distributed learning method. On the contrary, these internal rewards, on the other hand, make the learning effect worse than the original reward, *Simple Reward*.

Because the tasks in our test environment are relatively easy for the agent, that is, it can be done in a relatively short time. Therefore, we suppose that the original simple rewards can provide more intrinsic relationships or ways to improve learning effect for learning methods. The constructed internal rewards should be able to give clear guidance to the learning process and improve the speed of convergence of the learning methods. However, these forms of construction are subject to human factors. The constructed internal rewards method is not necessarily optimal. Therefore, we can conclude that the most primitive simple reward form has the best learning effect when the multi-agent task is relatively simple. Some of the constructed internal rewards are not necessarily the best, because of the specific guidance given by researchers of the specific learning effect may limited the independent improvement of learning effects.

However, since both sides agents are learning in our multi-agents adversarial environment, our above conclusions can only prove that when both sides use the same reward form and learn their own strategies, they use the more complex reward forms, *Both Speed Type Reward* and *Both Acceleration Type Reward*, compared with the original reward information, *Both Simple Reward*, the final strategy learning effect is worse for the predator during the same training episodes, but for the prey, it has better effect. It also means that it cannot be ruled out that due to the use of more complex forms of rewards, the improvement of the strategies learned by the prey is better than the improvement of the strategies by the predators, resulting in fewer times for the final predators to successfully pursue the prey. In order to verify our conjecture, we have tested and evaluated *Prey Use Simple Reward and Predator Use Speed Type Reward* and *Prey Use Speed Type Reward and Predator Use Simple Reward*. Table 2 and Figures 2, 5 and 6 show when the predator or the prey uses the simple reward form *Simple Reward*, it has a better strategy learning effect relative to the other party. And the other party that uses the more complicated form of reward, *Speed Type Reward,* has a worse strategy learning effect relative to the simple reward form *Simple Reward*. Therefore, we have ruled out our conjecture in this paragraph and verified the conclusion we proposed in the previous section.

VI. Conclusions And Future Work

The construction of reward functions is of great significance in the application of multi-agent reinforcement learning methods. A good internal construction reward form can solve the problem of poor strategy learning and slow convergence due to reward sparseness and delay in the strategy training process. In order to further evaluate the impact of different reward forms on the multi-agent deep reinforcement learning method strategy learning in a simple multi-agent collaboration and confrontation environment, to provide a guidance for the internal reward form construction of such methods, we use different forms of rewards in the same cooperation and competitive multi-agent environment in the distributed multi-agent reinforcement method MADDPG, and the learning effects of these rewards are evaluated. The results show that when the multi-agent coordination and confrontation tasks are relatively simple, the most primitive simple reward form has the best strategic learning effect. Some of the constructed internal rewards are not necessarily the best because of the specific guidance given by researchers of the specific learning effect may limited the independent improvement of learning effects. The results show that not all internal rewards can improve the learning effect of multi-agent reinforcement learning, which may be related to the complexity of the task and the training network structure. In our experiments, it was found that when the learning task is not very complicated, and the time for the team to complete the task is not very long, the learning effect of "Victory"-sparse rewards is the best, and the learning effects of the other two forms of reward are not as good as the learning effect of simple reward.

Because the training effects of different reward forms may be related to training methods and environment, and we conduct strategic training and evaluation in a single method and environment. Therefore, our next step will be to test the form of our rewards in more training methods and multi-agent collaboration and confrontation environments, especially in more complex environments to test strategies to verify our speculation and carry out more summary of the universal law. The effect of strategy learning may also be related to the weight coefficient of each component of the reward, and we will conduct more tests under the weight coefficient in the future. In addition, our different reward forms are based on the MADDPG method. In the future, we will conduct a research on the distributed multi-agent reinforcement learning strategy training method that adapts to our constructed reward form. It is worth noting that in the complex multi-agent collaboration and confrontation environment, such as the partial observable environment, the influence of the structure of the reward function form on the learning of the strategy deserves further study.

Acknowledgment

The author is very grateful to the people who gave valuable feedback during the writing process and the anonymous reviewers for their highly insightful comments and feedback.

References

[1] B. Siciliano, and O. Khatib, *Springer Hand book Of Robotics*, 2nd ed., Springer, 2016.

[2] R. Lowe, Y. Wu, and A. Tamar, et al, "Multi-agent actor-critic for mixed cooperative-competitive environments," *31st Conference on Neural Information Processing Systems(NIPS 2017)*, Long Beach, CA, USA, 2017.

[3] S. Wender, and I. Watson. "Applying reinforcement learning to small scale combat in the real-time strategy game StarCraft: Broodwar," *2012 IEEE Conference on Computational Intelligence and Games (CIG),* pp. 402-408, 2012.

[4] M. Kempka, M. Wydmuch, and G. Runc, et al, "ViZDoom: A Doom-based AI research platform for visual reinforcement learning," *IEEE Computational Intelligence and Games*, pp. 1-8, 2017.

[5] N. Usunier, G. Synnaeve, and Z. Lin, et al, "Episodic exploration for deep deterministic policies: an application to StarCraft micromanagement tasks," *Eprint ArXiv*, 2016.

[6] P. Peng, Y. Wen, and Y. Yang, et al, "Multiagent bidirectionally-coordinated nets: emergence of human-level coordination in learning to play StarCraft combat games," *Eprint ArXiv*, 2017.

[7] J. Foerster, G. Farquhar, and T. Afouras, et al, "Counterfactual multi-agent policy gradients," *The Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*, 2018.

[8] O. Vinyals, T. Ewalds, and S. Bartunov, et al, "StarCraft II: a new challenge for reinforcement learning," *Eprint ArXiv*, 2017.

[9] M. Lanctot, V. Zambaldi, and A. Gruslys, et al, "A unified game-theoretic approach to multiagent reinforcement learning," *The Thirty-first Annual Conference on Neural Information Processing Systems (NIPS)*, 2017.

[10]J. Max, M. C. Wojciech, and D. Iain, et al, "Human-level performance in first-person multiplayer games with population-based deep reinforcement learning," *Eprint ArXiv*, 2018.

[11]I. Mordatch, and P. Abbeel, "Emergence of Grounded Compositional Language in Multi-Agent Populations," *Eprint ArXiv*, 2017.