The Collaborative Combat of Heterogeneous Multi-UAVs Based on MARL

Jiayong Fang1,2, Yun Han2,*, Zhongliang Zhou1, Shitao Chen1 and Sheng Sheng1

1Equipment Management and Unmanned Aerial Vehicle Engineering College, Air Force Engineering University Xi’an 710051, China
2Science and Technology on Electro-optic Control Laboratory, LuoYang 471000, China

*Corresponding author email: fjylike@163.com

Abstract. In the course of collaborative combat of multi-UAVs, there appear so many unpredictable fluctuations in the combat situation, and the mission assignment and decision-making of the combat are sequential and game-oriented, rendering it hard to build its dynamic and accurate model of operational missions. This paper breaks through the traditional research idea of mission modeling and optimization solution of collaborative operations of UAVs, and applies multi-agent reinforcement learning to the decision-making of multi-UAV collaborative combat, builds a verification environment of collaborative air-to-ground penetration and attacks of heterogeneous UAVs by designing MARL algorithm architecture, situation model, decision model and reward and punishment model, realizes the autonomous evolutionary learning of mission assignment and decision-making, and finally analyzes and evaluates the effectiveness and robustness of different MARL algorithms.

1. Introduction

As UAV and artificial intelligence technology develop in leaps and bounds, distributed collaborative combat of UAVs will become a significant form of combat in the future. UAV collaborative combat means that multiple homogeneous or heterogeneous UAVs fulfill mission assignment and collaborative decision-making by means of mutual situational awareness and information sharing, so as to accomplish operational missions in conjunction. The traditional solution is to build a model for the problem of collaborative combat of multi-UAVs, including path planning or mission assignment, transform the planning and decision-making problem of operational missions of UAVs into the problem of multi-target optimization and decision-making with multiple constraints and strong coupling, and finally get the solution by theoretical algorithm of multi-target optimization. For instance, Document [1] builds a model for the problem of collaborative mission assignment of multi-UAVs as a problem of distributed constrained optimization, and Document [2] transforms the problem of reconnaissance path planning of multi-UAVs into a multi-target optimization model, and solves the optimal path by way of particle swarm optimization.

Then, it is pretty hard to build a decision-making model of multi-target optimization for the real collaborative combat process of UAVs. Primarily, the real combat process and combat situation are undergoing a myriad of changes in the twinkling of an eye, and it is difficult to have a command of complete combat information to build a mission assignment and decision-making model for collaborative combat before or during combat. Previous research and modeling are all based on certain assumptions, but this premise is difficult to be established in the actual combat process; besides, the
collaborative combat process of UAVs is game-based and sequential, that is, the mission assignment or decision-making model of one party's collaborative combat will be influenced by the other party's action model, and moreover, its current decision-making will influence the decision-making for the next moment, thus affecting the final mission assignment and decision-making results, while the traditional multi-target optimization and decision-making model of collaborative combat of UAVs does not take into account the game-based and sequential nature of decision-making.

Multi-agent reinforcement learning algorithm applies reinforcement learning technology, game theory and the like to multi-agent system, which enables multi-agents to perform complex collaborative missions via interaction and decision-making in the high-dimensional dynamic game process [3-5]. In 1990s, Littman[6] came up with MARL with Markov Decision Process (MDP) as the environmental framework, providing a simple and clear mathematical framework for addressing most problems related to reinforcement learning. Later, most researchers made further research on the basis of this model. Over these years, thanks to the success of deep learning, many deep reinforcement learning algorithms have been formed by combining deep learning methods with traditional reinforcement learning algorithms, which makes the research and application of single-intelligent agent reinforcement learning develop in a rapid manner. For example, AlphaGO, the Go game system developed by Deep-Mind Company has already defeated the top human players in the field of Go, and won with enormous advantages, which has tremendously shocked all sectors of society and stimulated researchers to devote more energy to the field of multi-agent intensive learning [7][8]. Enterprises represented by Deep-Mind, OpenAI as well as many universities have successively developed new MARL algorithms and applied them to real life. Currently, they are chiefly used in robot system [9-10], man-machine game [11-13], automatic driving [14], Internet advertising [15] and resource utilization [16-17].

In order to solve the problem of attack-defense countermeasure of large-scale unmanned aerial vehicle (UAV) swarm, [18] propose an improved Multi-agent algorithm (Multi-agent Proximal Policy Optimization, M-PPO) based on proximal policy optimization algorithm( PPO).

In this paper, the collaborative air-to-ground attack of heterogeneous UAVs is taken as the research object, and the model of combat rules is built as well, including maneuver model, jamming model, attack model and air defense model. By studying the multi-agent reinforcement learning algorithm model which is suitable for the mission decision of collaborative combat of heterogeneous UAVs, including MADDPG, MATD3 and M3DDPG algorithms, taking penetration and reconnaissance attacks by four heterogeneous UAVs (2 reconnaissance-attack UAVs, 2 jamming UAVs) against four moving threat sources and five moving targets as the simulation verification environment, this study verifies and evaluates the mission assignment and decision planning of three MARL algorithms in the collaborative air-to-ground attack of heterogeneous UAVs by designing their situation model, decision model and reward and punishment model. The simulation results indicate that after 1000 times of episode simulation, MADDPG, MATD3 and M3DDPG can attain the stable convergence phase, which enables four heterogeneous UAVs to effectively detect and strike moving targets with relatively small threats and better paths.

2. Scenario Description of Collaborative Air-to-ground Attack

4 UAVs are randomly initialized, including 2 reconnaissance-attack(RA) UAVs and 2 jamming(JAM) UAVs, and classify the reconnaissance-attack UAVs into two types, Type I and Type II. The initial information includes the information of coordinates X, Y, Z and the current heading angles of the aircraft (the speed is fixed, the pitch angle and roll angle are initially 0), four threat sources are initialized, including the information of coordinates X, Y, Z, the threat probability of each threat source decreases linearly with the distance from the center, and jamming UAVs can decrease the threat probability of the threat sources. Five targets are randomly initialized, and the targets are classified into two types, Type I and Type II, among which Type I targets can only be scoured and attacked by Type I reconnaissance-attack UAVs, and Type II targets can only be scoured and attacked by Type II reconnaissance-attack UAVs. By configuring the number ratio of Type I and Type II UAVs, and planning the action path of UAVs, UAVs can achieve reconnaissance and attack on targets with the least threat and the shortest time and path, as displayed in Figure 1.
3. Model and Algorithm of Collaborative Air-to-ground Attack

3.1. MADRL Algorithm

3.1.1. MADDPG Algorithm. Multi-Agent Deep Deterministic Policy Gradient (MADDPG) is an algorithm premised on AC framework on the principle of centralized training and decentralized execution. In the algorithm, each agent is provided with a centralized Critic to receive the information of other agents (such as actions and observations, etc.), namely:

\[ Q^\mu(o_1, a_1, o_2, a_2, ..., o_N, a_N) \]

In the meanwhile, the Actor network of each agent only executes the strategy \( a_i = \mu^\mu_i(o_i) \) based on its own partial observation, and the gradient of Critic network of each agent is up to the following:

\[ \nabla_{\theta_i} Q^\mu_i(x, a_1, a_2, ..., a_N) \big|_{a_i = \mu_i(o_i)} \]

The algorithm acquires the optimal strategy by continuously optimizing the loss function:

\[ L(\theta) = E_{x,a,r,x'}[(Q^\mu_i(x, a_1, a_2, ..., a_N) - y)^2] \quad y = r_i + \gamma Q^\mu_{j'}(x', a'_1, a'_2, ..., a'_N') \big|_{a_j = \mu_j(o_j)} \]

The algorithm renders it unnecessary to build the displayed communication rules, and besides, it is suitable for collaborative, competitive, mixed and a variety of other environments, which can well address the non-stationary problem of multi-agent environment.
As a matter of fact, the training process of each UAV is similar to that of a single DDPG algorithm, and the difference is chiefly embodied in the input of Critic: in the DDPG algorithm of a single UAV, the input of Critic is a state-action pair of information, but in MADDPG, the input of Critic of each UAV can be provided with additional information besides its own state-action information, such as the movements of other UAVs. Figure 2 is the network structure and training flow chart of MADDPG algorithm. It can be observed that each DDPG algorithm is faced with the current actor network and target actor network in entirely the same structure, as well as the current critic network and target critic network. Only the current actor network and the current critic network require to learn to update parameters, and both the target actor network and the target critic network copy the parameter values of the current network to the target network by means of soft update after the current network iterates for a period of time.

The environmental system makes the following assumptions:
(1) The strategy of each agent is merely subjected to its own observed state, but has nothing to do with the observed state of other agents, that is: \[ a_i = \mu_{i0}(o_i); \]
(2) The environment is unknown and model-free. The reward of each agent and the next state after taking action are unpredictable. The reward is attributable to the feedback of the environment, and its own action is only determined by the strategy;
(3) During training, the communication among all agents is not set, that is, they do not communicate with each other, or the content of communication is considered as a component of their observed value.

3.1.2. M3DDPG Algorithm. Minimax Multi-Agent Deep Deterministic Policy Gradient (M3DDPG) is an improved version of MADDPG algorithm. Despite that the stability of training can be guaranteed using a centralized critic, the learning strategy is still fragile and sensitive to other agents, and will converge to a bad local optimal state. Especially in the competitive environment, these problems will particularly come to the fore. Thus, on the basis of MADDPG algorithm, M3DDPG algorithm introduces minimax target function and multi-agent confrontation learning to enhance the robustness of learning strategy.

Minimax is a basic idea in zero-sum game. M3DDPG introduces minimax optimization into the function of learning target, which is intended to learn minimax strategy. The classical reinforcement learning is targeted at:
\[
\max_{\theta} \mathbb{E} \left[ \sum_t r_t(s_t, \pi(s_t|\theta)) \right] = \max_{\theta} \mathbb{E} [Q(s_0, \pi(s_0|\theta))] 
\]

Wherein,
\[ Q(s, a) = r(s, a) + \gamma Q(s', a') \]

While in M3DDPG, the learning target is:
\[
\max_{\theta} \min_{a_0, \ldots, a_T} \mathbb{E} \left[ \sum_t r_t(s_t, \pi_t(s_t|\theta_t), \ldots, a_j, \ldots) \right] = \max_{\theta} \min_{a_0, \ldots, a_T} \mathbb{E} [Q^M(s_0, \pi_i(s_0|\theta_i), \ldots, a_j, \ldots)] 
\]

Wherein,
\[ Q^M(s, a_i, \ldots, a_j, \ldots) = r(s, a_1, \ldots) + \gamma \min_{a'_{j+1}} Q^M(s', a'_{j+1}, \ldots) \]

The most significant thing in M3DDPG is how to deal with the problem of minimizing internal circulation in minimax target function. Minimizing internal circulation is a thorny problem, and besides it will cost much in calculation. Thus, M3DDPG further introduces multi-agent confrontation learning. The core idea of MAAL is to replace minimization of internal circulation with a single step gradient descent.
\[ Q^M(s, a_i, a_j) = r(s, a_i, a_j) + \gamma Q^M(s', a_i', a_j') \]

Wherein,
\[ a_j^{r(*)} = \min_{a_j} Q^M(s', a_i', a_j) \]

After optimization,
\[ a_j^{r(*)} \leftarrow a_j^{r(k+1)} = a_j^{r(k)} - \alpha \nabla_{a_j} Q^M(s', a_i', a_j) \big|_{a_j = a_j^{r(k)}} \]

Single-step gradient descent:
\[ a_j^{r(*)} = \pi_j(s') - \alpha \nabla_{a_j} Q^M(s', a_i', a_j) \big|_{a_j = \pi_j(s')} \]

Like MADDPG, M3DDPG resorts to centralized training and distributed execution (CTDE, Centralized Training, Distributed Execution), assuming that agents are entirely collaborative and can be controlled via a manager; its structure is exhibited in Figure 3. A centralized training module is used, with shared rewards, and it can be separated from each strategy throughout the execution process. On the strength of this centralized training module, M3DDPG resolves the unsteady problem in multi-agent reinforcement learning and furnishes global information for decentralized execution. By contrast, CTDE-based method is also faced with quite a number of limitations. Firstly, the scalability is relatively poor and the sample efficiency is comparatively low; moreover, in numerous other multi-agent missions, there are also semi-collaborative agents, and a semi-collaborative agent naturally has its own individual reward function (shared reward scenario is a special case).

**Figure 3.** CTDE structure diagramMATD3 algorithm

The MATD3 algorithm extends the TD3 algorithm for single agent training to the multi-agent area, which is, to some extent, similar to MADDPG extending DDPG to the multi-agent area. Thus, like MADDPG, MATD3 takes the structure of centralized training and distributed execution (CTDE), assuming that one agent can access the past movements, observations, rewards and strategies of all other agents during training.

Multi-Agent Twin Delayed DDPG, MATD3 presents a dual centralized critic network to decrease the Q value deviation.

Owing to the approximation error of MLP as well as its combination with strategy gradient, DDPG algorithm tends to overestimate the function of action value, resulting in slow convergence. Two trimmed Q networks and two target networks are used in the TD3 and DDPG algorithm is improved, so that it is able to deal with the problem of overestimation deviation of Q function in DDPG, and form the target in Berman error loss function with the smaller of the two Q values, which is similar to Double Q-learning to some extent.

Apart from that, TD3 updates the policy (and target network) less frequently than Q network, and it will update the policy and target network parameters only after updating Critic network d times. In the end, TD3 adds noise to the target action, which renders it more difficult for the strategy to make use of the error of Q function by smoothing the change of Q in the action.
**Smoothing of target policy:** the actions used to form Q-learning targets are based on target policy \( \mu_{\text{target}} \), but shear noise is added to every dimension of the actions. After adding shear noise, the target action will be cut into the effective operation range (all effective actions \( a \) meet the condition: \( d_{\text{low}} \leq a \leq d_{\text{high}} \)). Thus, the target action becomes:

\[
a'(s') = \text{clip}(\mu_{\text{target}}(s') + \text{clip}(\epsilon, -c, c), d_{\text{low}}, d_{\text{high}}), \quad \epsilon \sim N(0, \delta)
\]

Smoothing of target policy is essentially the regularization of the algorithm. It resolves the specific failure modes that may occur in DDPG: if the Q-function approximator forms an incorrect peak for some actions, the strategy will quickly make use of the peak, and then fragile or incorrect behaviors will occur. This situation can be avoided by smoothing the q function, and the smoothing of target strategy is to smooth similar actions. The similar mechanism has also been used in network attacks\(^{[19]}\).

**Trimmed double Q network:** both q-functions use one target, and the target value calculated by either of the two q-functions is smaller:

\[
L(\phi_1, D) = E_{(s,a,r,s',d)}[(Q_{\phi_1}(s,a) - y(r,s',d))^2],
\]

\[
L(\phi_2, D) = E_{(s,a,r,s',d)}[(Q_{\phi_2}(s,a) - y(r,s',d))^2].
\]

Using a smaller Q value as the target value, and then regressing the value step by step, is helpful to avoid overestimation of Q function.

The strategy network is almost no different from DDPG by maximizing \( Q_{\phi} \) learning:

\[
\max_{\phi} E_{s \sim D}[Q_{\phi}(s, \mu_{\phi}(s))].
\]

However, in TD3, the update frequency of strategy is lower than that of Q function. This helps to decrease the volatility that usually occurs in DDPG owing to how the policy update changes the target.

### 3.2. Simulation Model

#### 3.2.1. Maneuver Model

The default condition is that the angle of attack and sideslip angle are both 0, that is, the airframe coordinate system is consistent with the track coordinate system.

Constraints: maximum slope (roll angle): \( \gamma_{+} \) and \( \gamma_{-} \) (generally not more than 30 degrees); maximum overload: \( n_{+} \) and \( n_{-} \), where \( \theta \) is pitch angle and \( \phi \) is yaw angle.

1. Change course: the aircraft keeps high stability, turns and turns without sideslip, and the yaw rate of the aircraft is:

\[
\dot{\phi}(t) = -\frac{g}{v(t)\cos\gamma(t)}\sin\gamma(t)
\]

Wherein, when \( a(t) = lt \), \( \gamma(t) = \gamma_{+} \), and when \( a(t) = rt \), \( \gamma(t) = \gamma_{-} \), then:

\[
\begin{align*}
\phi(t+1) &= \phi(t) + \Delta t \phi(t) \\
x(t+1) &= x(t) + \Delta t v(t) \cos \theta(t) \cos \phi(t) \\
z(t+1) &= z(t) + \Delta t v(t) \cos \theta(t) \sin \phi(t) \\
y(t+1) &= y(t) + \Delta t v(t) \sin \theta(t)
\end{align*}
\]

2. Change altitude: the aircraft keeps the current heading and raises the pitch rate of altitude:
\[ \dot{\theta}(t) = \frac{g}{v(t)}(n - \cos \theta(t)) \]

\[
\begin{align*}
\theta(t+1) &= \theta(t) + \Delta t \left( \frac{g}{v(t)}(n - \cos \theta(t)) \right) \\
x(t+1) &= x(t) + \Delta v(t) \cos \theta(t) \cos \varphi(t) \\
z(t+1) &= z(t) + \Delta v(t) \cos \theta(t) \sin \varphi(t) \\
y(t+1) &= y(t) + \Delta v(t) \sin \theta(t)
\end{align*}
\]

When \( a(t) = up \), \( n = n_+ \), and when \( a(t) = dn \), \( n = n_- \)

(3) Stable flight
The aircraft keeps its speed, track angle and heading angle unchanged.

\[
\begin{align*}
x(t+1) &= x(t) + \Delta v(t) \cos \theta(t) \cos \varphi(t) \\
z(t+1) &= z(t) + \Delta v(t) \cos \theta(t) \sin \varphi(t) \\
y(t+1) &= y(t) + \Delta v(t) \sin \theta(t)
\end{align*}
\]

3.2.2. Threat Model. (1) The maximum threat distance is 300 kilometers, and the threat probability at this time is 0.3;
(2) Following the shortening of the distance center, the threat probability increases linearly, that is, from 300-30 to 0.3-1;
(3) In areas where multiple threat areas overlap, the threat probability should be increased. If the threat probability of threat source 1 is \( P_1 \), if the threat probability of threat source 1 is \( P_2 \), the threat probability after overlapping is \( 1-(1-P_1)(1-P_2) \).

3.2.3. Jamming Model. (1) The maximum jamming distance is 150km;
(2) In the case of 150 km jamming, the threat probability can be attenuated by 10%, and the maximum attenuation rate is 50%, that is, the attenuation ratio corresponding to (150-30) is (10%-50%);
(3) The attenuation coefficient increases by 1.1 times for each additional jamming aircraft.

4. Simulation Experiment of Collaborative Air-to-ground Attack

4.1. Situation Information
The relative situation information between UAVs is presented in Figure 4, including the situation information between UAVs and the relative position information between UAVs and targets.

![Figure 4](image_url)

**Figure 4.** Relative situation information of UAVs
The input situation information includes local position coordinates (X, Y, Z), local heading (pitch and yaw), local sensor types, other UAV position coordinates (X, Y, Z) and heading information (pitch and yaw), as well as their sensor types, threat source position coordinates (X, Y, Z), target position
coordinates (X, Y, Z), and relative situation information, such as relative distance, relative altitude, relative interval, target angle, and entry angle.

4.2. Decision Information
There are five basic maneuver rule sets, namely, turning left, turning right, jumping, diving and keeping the current motion state, as well as the combination forms of different UAV types. It can also directly output continuous roll angle or overload as decision rule set.

4.3. Reward and Punishment Information
A certain punishment will be imposed at each deduction step of UAVs, and a certain reward will be awarded for each suitable target detected, or the closer it is to the target, the greater the threat and punishment, and the longer it lasts in the threat source, the greater the threat and the more punishment.

4.4. Simulation Results
The combat process of collaborative air-to-ground attack of heterogeneous UAVs is demonstrated in Figure 5, and the convergence of the algorithm is displayed in Figure 6. The hyper-parameters of MADDPG, M3DDPG and MATD3 are set as below: learning rate is 0.01, Batch-size is set to 1024, \( \tau = 0.01 \); for all Q-value networks and policy networks, the network structure is two hidden layers with 400 neurons and 300 neurons respectively. The activation function adopts ReLU, while the output layer uses sigmoid to activate. Then the output is linearly scaled to the corresponding motion space range.

In Figure 6, the abscissa represents episode in K, and the ordinate means the average reward value per 1000 episode. It can be learned that the initial reward value of M3DDPG is the highest, followed by MADDPG, while MATD3 is the lowest. Besides, the growth rate of rewards gained by the three algorithms in this scenario is also M3DDPG>MADDPG>MATD3, and finally, the reward will be stable at around 0. In Figure 7, the average time spent per episode is MATD3>MADDPG>M3DDPG. In Figure 8, the normalized final team award is shown as M3DDPG>MATD3>MADDPG>DDPG. It
is suggested that M3DDPG is the best with regard to time consumption and reward value, while MADDPG is superior in time consumption and MATD3 is superior in final reward in this distributed air combat scenario of semi-confrontation and semi-collaboration.

5. Conclusion

The traditional research idea to address the problem of collaborative combat of multi-UAVs is to transform the collaborative combat mission into a multi-target optimization decision model to get the solution. However, the actual combat process carries the dynamic, random, uncertain and game-based characteristics, etc., so it is hard to build an accurate model for mathematical description. In this paper, multi-agent deep reinforcement learning is applied to the collaborative combat process of multi-UAVs, rendering it unnecessary to build a mathematical model for the decision-making of mission assignment of collaborative combat of multi-UAVs. By designing combat rules, situation sets, reward and punishment sets and action sets, and using MARL algorithm architecture, self-learning and self-exploration for the decision-making of combat missions can be realized, and its solution is more in line with the actual combat process. At last, the effectiveness and robustness of the three MARL algorithms in collaborative combat of UAVs are verified in the verification environment of collaborative air-to-ground attacks of multi-UAVs. The simulation results reveal that the MARL algorithm can be effectively applied in the mission decision-making process of collaborative combat of UAVs, and boasts enormous potential in the future collaborative combat of multi-UAVs.

References

[1] Wu Ling, ZHANG Zhu-feng, WU Wei. Coordinated test allocation in multi-UCAV based on distributed constrained optimization[J].Journal of Naval University of Engineering,2018,12(6) :64-68.
[2] Hu Teng, LIU Zhangjun, LIU Yang, et al. 3D surveillance path planning for multi-UAVs[J]. Systems Engineering and Electronics[J].2019,7(7):1551-1559.
[3] Lucian Bu, Robert Babu, Bart De Schutter, et al. A comprehensive survey of multiagent reinforcement learning. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 2008, 38(2):156–172.
[4] Lætitia Matignon, Laurent Jeanpierre, and Abdel-Illah Mouaddib. Coordinated multi-robot exploration under communication constraints using decentralized markov decision processes. In Twenty-sixth AAAI conference on artificial intelligence, 2012.
[5] Peng Peng, Quan Yuan, Ying Wen, Yaodong Yang, Zhenkun Tang, Haitao Long, and Jun Wang. Multiagent bidirectionally-coordinated nets for learning to play starcraft combat games. arXiv preprint arXiv:1703.10069, 2, 2017.
[6] LITTMAN M L. Markov games a framework for multi-agent reinforcement learning[M]. New Brunswick: Machine Learning Proceedings, 1994:157-163.
[7] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Der Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershel-vam, Marc Lancotot, et al. Mastering the game of go with deep neural networks and tree search. nature, 529(7587):484, 2016.
[8] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lancotot, Laurent Sifre, Dharshan Kumaran, Thore Grae-pel, et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. arXiv preprint arXiv:1712.01815, 2017.
[9] Gu S, HOLLY E, LILLICRAPP T, et al. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates[C]//IEEE International Conference on Robotics and Automation. Singapore: IEEE Press, 2017:3389-3396.
[10] FOERSTER J, ASSAIL I, DE FREITAS N, et al. Learning to communicate with deep multi-agent reinforcement learning[C]//Advances in Neural Information Processing Systems. Spain: NIPS Press,2016:2137-2145.
[11] LOWE WU Y, et al. Multi-agent actor-critic for mixed cooperative-competitive Environments[C]/advances in Neural Information Processing Systems. Los Angeles: NIPS Press, 2017:6379-6390.

[12] LANCTOT M, ZAMBA LD V, GRUSLYS A, et al. A unified game-theoretic approach to multi-agent reinforcement learning[C]/Advances in Neural Information Processing Systems. Los Angeles: NIPS Press, 2017:4190-4203.

[13] LEIBO J, ZAMBA LD V, LANCTOT M, et al. Multi-agent reinforcement learning in sequential social dilemmas[C]/Proceedings of the 16th Conference on Autonomous Agents and Multi-agent Systems, Singapore: AAMAS Press, 2017:464-473.

[14] SHALEV-SHWARTZ S, SHAMMAH S, SHASHUA A. Safe, multi-agent reinforcement learning for autonomous driving [J].https://arxiv.org/abs/1610.03295.

[15] JIN J, SONG C, LI H, et al. Real-Time Bidding with Multi-Agent Reinforcement Learning in Display Advertising[J/OL].https://arxiv.org/abs/1802.09756.

[16] XI L, CHEV J, HUANG Y, et al. Smart generation control based on multi-agent reinforcement learning with the idea of the time tunnel[J].Energy, 2018, 153:977-987.

[17] PEROLAT J, LEIBO JZ, ZAMBA LD V, et al. A multi-agent reinforcement learning model of common-pool resource appropriation[C] Advances in Neural Information Processing Systems. Los Angeles: NIPS Press, 2017:3643-3652.

[18] XUAN Shuzhe, KE Liangjun. Study on Attack-Defense Countermeasure of UAV Swarms Based on Multi-agent Reinforcement Learning[J]. Radio Engineering, 2021, 51(5): 360-366.

[19] Shang, Yilun. Robustness of scale-free networks under attack with tunable grey information[J]. EPL (Europhysics Letters), 2011, 95(2):28005.