IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 30, 2019, accepted September 12, 2019, date of publication September 23, 2019, date of current version October 21, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2943253

Joint Optimization of Multi-UAV Target Assignment and Path Planning Based on Multi-Agent Reinforcement Learning

HAN QIE[1], DIANXI SHI[2,3], TIANLONG SHEN[2], XINHAI XU[2], YUAN LI[2], AND LIUJING WANG[1]
[1]College of Computer, National University of Defense Technology, Changsha 410073, China
[2]Artificial Intelligence Research Center, National Innovation Institute of Defense Technology, Beijing 100071, China
[3]Tianjin Artificial Intelligence Innovation Center, Tianjin 300457, China

Corresponding author: Dianxi Shi (dxshi@nudt.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFB1001901, in part by the National Defense Science and Technology Foundation for Young Scientists of China under Grant 030403, and in part by the National Natural Science Foundation of China under Grant 11801563, Grant 91648204, Grant 61532007, and Grant 61902425.

ABSTRACT One of the major research topics in unmanned aerial vehicle (UAV) collaborative control systems is the problem of multi-UAV target assignment and path planning (MUTAPP). It is a complicated optimization problem in which target assignment and path planning are solved separately. However, recalculation of the optimal results is too slow for real-time operations in dynamic environments because of the large number of calculations required. In this paper, we propose an artificial intelligence method named simultaneous target assignment and path planning (STAPP) based on a multi-agent deep deterministic policy gradient (MADDPG) algorithm, which is a type of multi-agent reinforcement learning algorithm. In STAPP, the MUTAPP problem is first constructed as a multi-agent system. Then, the MADDPG framework is used to train the system to solve target assignment and path planning simultaneously according to a corresponding reward structure. The proposed system can deal with dynamic environments effectively as its execution only requires the locations of the UAVs, targets, and threat areas. Real-time performance can be guaranteed as the neural network used in the system is simple. In addition, we develop a technique to improve the training effect and use experiments to demonstrate the effectiveness of our method.

INDEX TERMS Multi-UAV, target assignment and path planning, multi-agent reinforcement learning, MADDPG, dynamic environments.

LIST OF ABBREVIATIONS

CPBAA	Collision Probability Between Agent and Agent
CPBAT	Collision Probability Between Agent and Threat
CWS	Cell Winnings Suppression
DDPG	Deep Deterministic Policy Gradient
DQN	Deep Q-learning Network
GAs	Genetic Algorithms
MADDPG	Multi-Agent Deep Deterministic Policy Gradient Algorithm
MARL	Multi-Agent Reinforcement Learning
MAS	Multi-Agent System
MDPs	Markov Decision Processes
MILP	Mixed-Integer Linear Programming
MLP	Multi-Layer Perception
MUTAPP	Multi-UAV Target Assignment and Path Planning
PP	Path Planning
PSO	Particle Swarm Optimization
RHTA	Receding Horizon Task Assignment
SDT	Satisfaction Decision Theory
STAPP	Simultaneous Target Assignment and Path Planning
TA	Target Assignment
TCR	Task Completion Rate
UAVs	Unmanned Aerial Vehicles

The associate editor coordinating the review of this manuscript and approving it for publication was Yichuan Jiang.

I. INTRODUCTION

Recent developments in unmanned aerial vehicles (UAVs) have made them indispensable tools in modern society. They

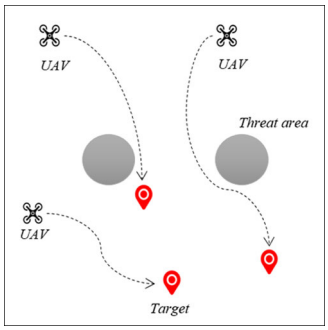


FIGURE 1. Multi-UAV targets assignment scenario.

can perform a variety of tasks previously done by humans in dangerous and complex environments, such as target searching, attacking, and so on [1]. For complicated tasks, cooperative formations of UAVs have become an important means of improving efficiency. The key to solving cooperative problems is multi-UAV target assignment and path planning (MUTAPP), which has recently attracted a great deal of attention.

Researchers have considered the solutions of MUTAPP in different scenarios and for different tasks [2]. In this paper, we consider the scenario shown in Fig. 1: (1) A formation of UAVs is required to fly to targets distributed at different locations with the shortest total flight distance. (2) There are some fixed threat areas the UAVs cannot enter. (3) Collision avoidance between UAVs is required. (4) There is only one type of target and all UAVs are considered identical. It is a complex combinatorial optimization problem that includes two main sub-problems: target assignment (TA) and path planning (PP). The TA is an optimization problem which has been proven to be NP-hard. Many methods have been studied to solve it, such as heuristic techniques [3], [4], network flow analogies [5], and market approaches [6], [7]. The PP problem has been solved by Voronoi diagrams with the Dijkstra algorithm [8], adaptive random search algorithm [9], model predictive control [10], and mixed-integer linear programming (MILP) [11], [12]. Liangheng Lv et al. proposed a method based on reinforcement learning [13].

These algorithms cannot deal with dynamic environments effectively because they require global environmental information to calculate the optimal result. However, real environments change rapidly. For example, the locations of the targets may move randomly. Once the environment changes, the original results will fail. Furthermore, the process of recalculating the optimal results is too slow for real-time operations because of the large number of calculations required [14]. These shortcomings limit the application of such algorithms in groups of UAVs in real, dynamic environments.

The development of multi-agent reinforcement learning (MARL) provides a new solution for MUTAPP problems. The multi-agent deep deterministic policy gradient (MADDPG) [15] algorithm is a state-of-the-art MARL algorithm that performs well in multi-agent collaborative,

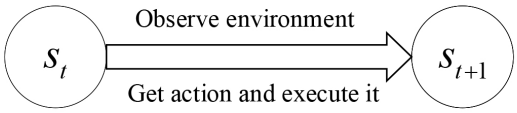


FIGURE 2. Execution process is regarded as MDPs.

competitive or mixed environments. The MUTAPP problem is essentially a mixed scenario of competition and cooperation. Competition means that each UAV must cover a target exclusively, while cooperation means that the UAV formation must minimize the total travel cost. In addition, the MADDPG model can deal with dynamic environments after training. The reason is that the environmental information is generated randomly in each training episode. Therefore, the ability of the model to adapt to a changing environment becomes stronger with an increasing number of training episodes. Based on the above ideas, we propose an intelligent method named the simultaneous target assignment and path planning (STAPP) algorithm, which is based on MADDPG. As far as we know, no such method exists in this field. The MUTAPP problem is incorporated into a multi-agent system (MAS), which is trained by a modified MADDPG in dynamic environments. The accomplished system only requires the location information of the environment for execution. Our method considers the MUTAPP problem as a Markov decision process (MDP). The UAVs' actions are discretized into time steps. After each step, the UAV formation and environmental conditions are considered as a state. The flight action each UAV chooses is only relevant to the current state. Fig. 2 shows the state transition process from s_t to s_{t+1} . Each UAV observes the current environment and then gets its next action from its own policy network. This process is described in detail in Part *B* of Section II. Furthermore, the policy network uses 2-layer 128-unit multi-layer perception (MLP) policies, which guarantees real-time operation.

The rest of this paper is organized as follows. Section II reviews related work, Section III provides a detailed introduction of our method, Section IV describes the experiments, and Section V summarizes our conclusions and future work directions.

II. RELATED WORK

A. MUTAPP PROBLEM

Researchers have done much work solving MUTAPP problems. Bellingham et al. proposed a mathematical approach with the flexibility to include more detailed constraints based on MILP [12]. However, it can only be solved by professional mathematical software as the solution is very complex. Then, Beard et al. proposed a method based on satisfaction decision theory (SDT), which constructs its own satisfaction set for each UAV and then searches for the global optimal allocation scheme for all individual satisfaction sets [16]. This reduces the search space but also degrades the optimality of the results. They also assumed that each UAV flies at a different

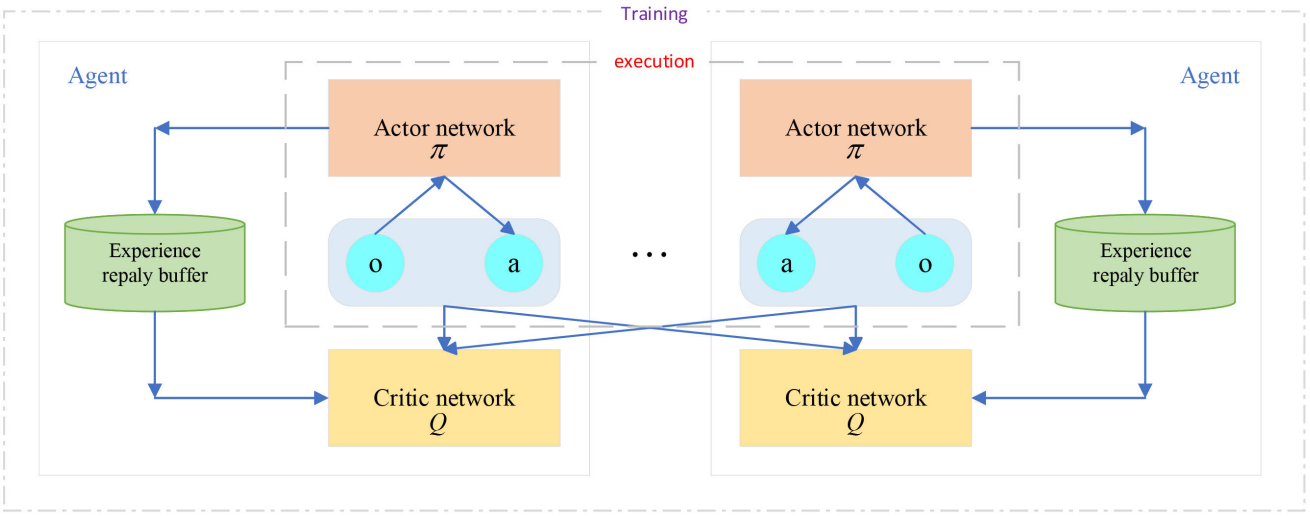


FIGURE 3. The framework of MADDPG.

preassigned altitude, thereby ensuring collision avoidance without the need to plan collision-free paths [2]. Shima et al. effectively solved the MUTAPP problem based on genetic algorithms (GAs) [17] which consider unique requirements such as task priority, coordination, time constraints, and flight trajectories. Cruz Jr et al. presented an algorithm based on the principles of particle swarm optimization (PSO) [18]. In recent years, Babel et al. addressed the problem of cooperative flight path planning under the condition that the UAVs arrive at their destinations simultaneously, or sequentially with specified time delays, while minimizing the total mission time [19]. Jia-lei et al. established a model of UAV attack advantage and target threat level based on cell-winning suppression (CWS) [20], which proved to be computationally efficient. These algorithms described above solve specific problems based on different conditions. However, they cannot deal with dynamic environments because they use global environmental information.

There is relatively little research in this field that considers dynamic environments. Bellingham et al. considered cooperative path planning for multiple UAVs in dynamic and uncertain environments by using MILP [21]. Alighanbari et al. proposed an algorithm named receding horizon task assignment (RHTA) for re-planning in dynamic environments [22]. Their drawback is that they deal with dynamic environments through rapid recalculation, which has a huge computational cost.

Unlike any of the above methods, our proposed method utilizes the powerful data representation and decision-making capabilities of deep reinforcement learning to effectively deal with dynamic environments via training of the system.

B. MADDPG FRAMEWORK

The MADDPG framework is shown in Fig. 3. Each agent has two networks: an actor network π and a critic network Q. The actor network calculates the action to be executed based on the state acquired by the agent, while the critic network evaluates the action calculated by the actor network to improve the performance of the actor network. An experience replay buffer is used to store a certain amount of training experience, which Q reads randomly when updating the network in order to break correlations in the training data and make the training more stable. In the training phase, the actor network only obtains observation information from itself, while the critic network acquires information such as the actions and observations of other agents. In the execution phase, the critic network is not involved, and each agent only needs an actor network. This means that the execution is decentralized.

Actually, MADDPG can be considered as a multi-agent version of DDPG (deep deterministic policy gradient) [23]. Its core idea is to concentrate training and decentralize execution. The Q-learning DQN (deep Q-learning network) [24] and DDPG perform poorly in multi-agent environments because they do not use the information of other agents [15]. The MADDPG approach overcomes this difficulty by using the observations and actions of other agents.

III. STAPP METHOD

In this section, we outline the ideas and details of our STAPP method. Firstly, the MUTAPP problem is formally described as an optimization problem. Then, we build a MAS based on the MADDPG framework to fit the optimization conditions. The system can solve TA and PP simultaneously and deal with dynamic environments due to changed environments in training. Next, the process of STAPP is given. Finally, we discuss the key technology and give the pseudo-code of STAPP.

A. FORMALIZATION OF THE MUTAPP PROBLEM

Consider a team of UAVs $U_i \in \mathbf{U}, i = 1, \ldots, N_U$ in a two-dimensional environment. The position of UAV U_i at time t

is given by $\left(x_i^U(t), y_i^U(t)\right)$. The targets $T_i \in \mathbf{T}, i = 1, \ldots, N_T$ and threats area $D_i \in \mathbf{D}, i = 1, \ldots, N_D$ are distributed across it. The position of target T_i is denoted by $\left(x_i^T, y_i^T\right)$, while threat D_i is given by $\left(x_i^D, y_i^D\right)$. For UAV, U_i, a path $\{P_i\}$ is planned and given by (1) and the length of $\{P_i\}$ is given by d_i.

$$P_i = \left\{\left(x_i^U(0), y_i^U(0)\right), \left(x_i^U(1), y_i^U(1)\right), \ldots, \left(x_i^U(n), y_i^U(n)\right)\right\} \quad (1)$$

The optimization goal of the MUTAPP is $\min\left(\sum_i^{N_U} d_i\right)$ and it must satisfy the following constraints:

1. Each target $T_i'$is assigned to one UAV,i.e., $\bigcup_{i=1}^{N_U} T_i' = \mathbf{T}, i \in \{1, \ldots, N_T\}$.

2. Each target is only assigned to one UAV, i.e., $\forall i \neq j, T_i' \neq T_j' \quad i, j \in \{1, \ldots, N_T\}$.

3. Each UAV's path is collision-free with threat area, i.e., $\forall i, j, D_j \notin P_i \quad i \in \{1, \ldots, N_T\}, j \in \{1, \ldots, N_U\}$.

4. Each UAV's path is collision-free with other UAVs, i.e., $\forall i, j, \left(x_i^U(t), y_i^U(t)\right) \neq \left(x_j^U(t), y_j^U(t)\right)$ at any time t.

B. BUILDING MAS BASED ON MADDPG

The model described in Part *A* shows that there are competition and cooperation between UAVs, which can be modeled by the MADDPG framework. We build a MAS to solve the model in this section instead of using traditional optimization algorithms. In this system, each UAV is abstracted as an agent, which has the same action model as a real UAV.

A scenario is considered with N_U agents using policies parameterized by $\theta = \left\{\theta_1, \theta_2, \cdots, \theta_{N_U}\right\}$, and let $\mu = \left\{\mu_{\theta_1}, \mu_{\theta_2}, \cdots, \mu_{\theta_{N_U}}\right\}$(abbreviated as μ_i)be the set of all agent deterministic policies. For the deterministic policy μ_i of agent i, the gradient can be written as (2).

$$\nabla_{\theta_i} J(\mu_i) = \mathbb{E}_{\mathbf{x}, a \sim \mathcal{D}}\left[\nabla_{\theta_i}\mu_i(a_i|o_i)\, \nabla_{a_i} Q_i^{\mu}(\mathbf{x}, a_1, \ldots, a_{N_U})\big|_{a_i=\mu_i(o_i)}\right] \quad (2)$$

where $\mathbf{x} = \left\{o_1, o_2, \ldots, o_{N_U}\right\}$ represent the state, $Q_i^{\mu}(\mathbf{x}, a_1, \ldots, a_{N_U})\big|_{a_i=\mu_i(o_i)}$ is the $Q-value$ function, a_i is the action of agent i and o_i is the observation of agent i, which includes the distance between the agent i and each target, the distance between the agent i and other agents, and the distance between the agent i and the threat area.

The $\mathcal{D}$ represents the experience replay [24] buffer which contains a series of tuples $\left(\mathbf{x}, \mathbf{x}', a_1, \ldots, a_{N_U}, r_1, \ldots, r_{N_U}\right)$, recording the experiences of all agents. The $\mathbf{x}'$ is the new state of the MAS after executing the actions and the r_i is the reward of agent i. The critic-network Q_i^{μ} is updated by the loss function written as (3).

$$\begin{aligned} \mathcal{L}(\theta_i) &= \mathbb{E}_{\mathbf{x}, a, r, \mathbf{x}'}\left[\left(Q_i^{\mu}(\mathbf{x}, a_1, \ldots, a_{N_U}) - y\right)^2\right] \\ \text{where } y &= r_i + \gamma Q_i^{\mu'}(\mathbf{x}', a_1', \ldots, a_{N_U}')\big|_{a_j'=\boldsymbol{\mu}_j'(o_j)} \end{aligned} \quad (3)$$

The actor-network is updated by minimizing the policy gradient of agent i which can be written as (4).

$$\nabla_{\theta_i} J \approx \frac{1}{S}\sum_j \nabla_{\theta_i}\mu_i\left(o_i^j\right) \nabla_{a_i} Q_i^{\mu}\left(\mathbf{x}^j, a_1^j, \ldots, a_{N_U}^j\right)\Big|_{a_i=\mu_i\left(o_i^j\right)} \quad (4)$$

where S is a random minibatch size of samples and j is the index of samples.

By building a MAS based on MADDPG, the MUTAPP is considered as a complete system. Therefore, the two subproblems of TA and PP can be solved simultaneously.

C. KEY TECHNOLOGY OF STAPP

In this subsection, we describe the key technology of STAPP. As mentioned above, the STAPP method solves both TA and PP problems simultaneously. The key to TA problems is minimizing the travel distance, while the key to PP problems is collision avoidance. Therefore, the appropriate reward structure needs to be designed to achieve these two optimization goals. In addition, a method of enhancing the training effect is proposed.

1) SHORTEST TRAVEL DISTANCE

Minimizing the travel distance has many important outcomes, such as saving time, electricity, and so on. Motivated by the cooperative navigation experiment in [15], we design a reward structure to accomplish goal assignment and minimize travel distance, as Algorithm 1 shows. After each step of the agent, the MAS traverses each target to find the nearest agent to it and adds the distance to reward after taking the opposite number. It is well known that the optimization goal of reinforcement learning algorithms is reward maximization, so the unified optimization goal is achieved by taking the opposite number.

Algorithm 1 Calculate Reward r_1

```
Initialize r_1 = 0
for target 0 to N_T do
    Calculate the distance set d of all agents to the target
    r_1 = r_1 + (−min(d))
end for
return r_1
```

2) COLLISION AVOIDANCE

Next, we consider collision avoidance. When a collision occurs, the agent will receive a negative reward. In order to improve the effect of collision avoidance, we naturally think of adding critical areas to threat areas and agents. As shown in Fig. 4, the critical areas are extensions of the original area. The original collision detection standard is that the distance between objects is less than the sum of their radius. After adding critical areas, when critical areas collide with each other the agent obtains a negative reward , which is equivalent to an early collision warning mechanism. In this way, a certain collision-avoidance reaction time can be assigned to the agent.

When two critical areas collide with each other, we call it a pseudo-collision. Similarly, when the agents collide with each other or with a threat area, we call it real-collision. In training, the number of real-collisions can be reduced by penalizing the occurrence of pseudo-collision. The experiment in Section IV shows the effect of critical area size on the experimental

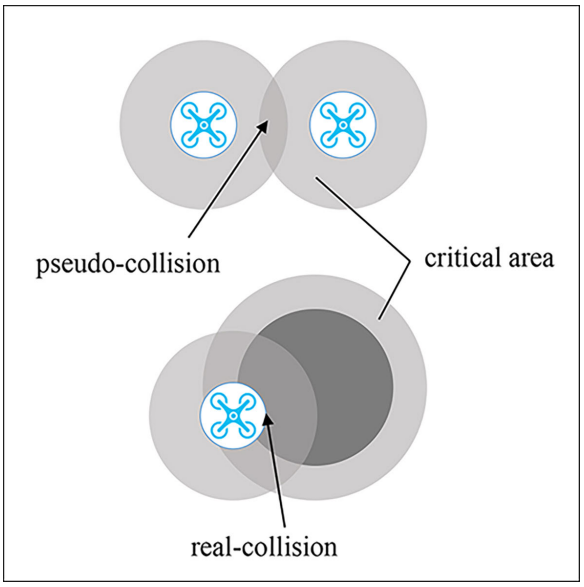


FIGURE 4. Critical area, pseudo-collision and real-collision.

indicators. The reward structure used to train the UAV collision avoidance is expressed as Algorithm 2.

Algorithm 2 Calculate Reward r_2

```
Initialize r2 = 0
for agent 0 to N_U do
    if dist2 < dist_min2 + σ then
        r2 = r2 − (dist_min2 + σ − dist2)
    else
        r2 = r2 − 0
    end if
end for
return r2
```

where $dist_{min2} = agent.size + agent.size$, σ represents the width of critical area of agents and $dist_2$ represents the distance between agents.

The third part of reward r_3 is to train the UAVs to avoid threat areas. The reward value is set as per Algorithm 3.

Algorithm 3 Calculate Reward r_3

```
Initialize r3 = 0
for agent 0 to N_U do
    if dist3 < dist_min3 + σ then
        r3 = r3 − (dist_min3 + σ − dist3)
    else
        r3 = r3 − 0
    end if
end for
return r3
```

where $dist_{min3} = agent.size + threatarea.size$, σ represents the width of critical area of threat areas and $dist_3$ represents the distance between the agent and threat areas.

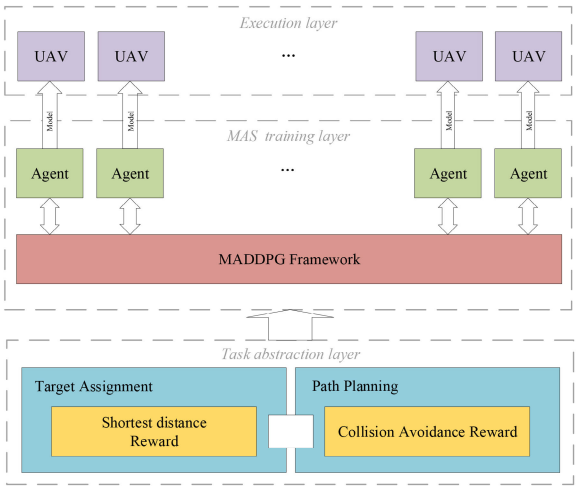


FIGURE 5. The structure of STAPP.

D. CONSTRUCTION OF STAPP

Now we can construct a complete STAPP method as shown in Fig. 5. It consists of three layers. Firstly, the task abstraction layer transforms the task optimization process into the corresponding reward structure convergence process. For the MUTAPP problem, the TA and PP sub-problems should be optimized simultaneously. The goal of TA is to minimize the travel distance of the UAVs. The goal of the PP problem is to minimize collisions as much as possible. They correspond to the two reward structures of Part *C* of the STAPP method.

The MAS training layer is based on the first level. It consists of a training environment and training algorithm. The environment, including agents (abstracted UAVs), targets, and threat areas, can be set as needed. The level of algorithm

Algorithm 4 STAPP

```
Initialize N_U, N_T, N_D, and σ
for episode = 1 to M do
    Receive initial state x
    for t = 1 to MAX-Step do
        for each agent i, select action a_i = μ_{θ_i}(o_i) + N_t
    w.r.t. the current policy and observation
        Execute actions and observe reward r = r1 + r2 +
    r3 and new state x'
        Store experience in replay buffer D
        for each agent i = 1 to N_U do
            Sample a random minibatch of S samples
    from D
            Update the Critic network by minimizing the
    loss as formula (3)
            Update the Actor network using the sampled
    policy gradient as formula (4)
        end for
        Update the target network for each agent i
    end for
end for
```

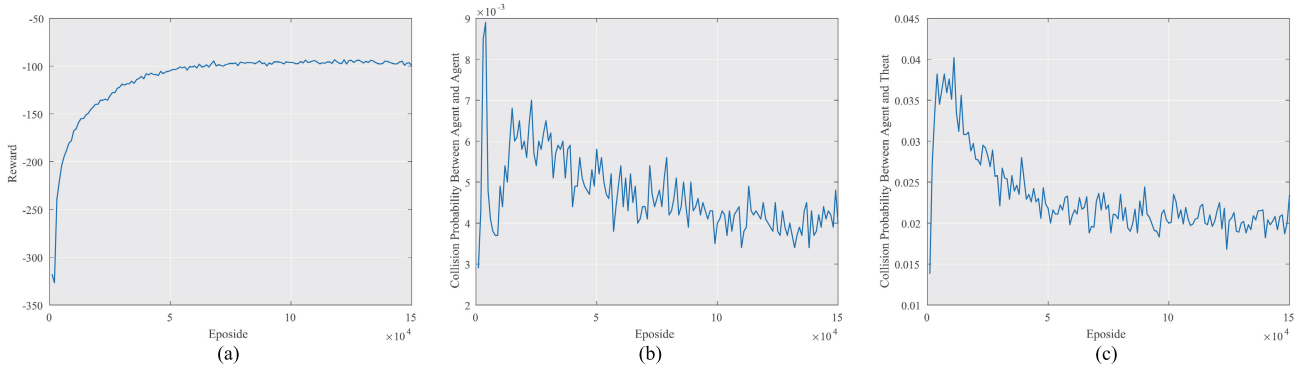


FIGURE 6. The Reward, CPBAA and CPBAT results obtained by original MADDPG.

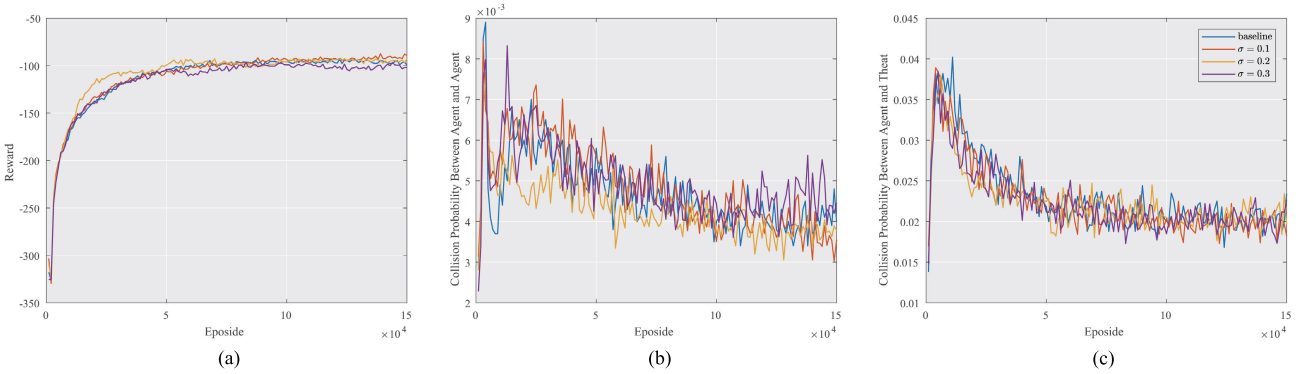


FIGURE 7. Comparison of the reward, CPBAA and CPBAT with $\sigma = 0, 0.1, 0.2, 0.3$.

includes but is not limited to MADDPG can be any algorithm suitable for multi-agent training. After training, each agent will get a policy, which is actually an actor network, which receives an observation value and outputs an action.

The top layer is the executive layer, where each agent's policy is deployed to a real UAV in the formation.

We can give the pseudo code of STAPP as Algorithm 4 shows.

Step 1 initializes the training environment. Steps 2-15 are the MADDPG framework process, as shown in Part *B* of this section. Symbol o_i repersents the observation of agent i and $\mathcal{N}_t$ is the random noise that occurs in training. Although the TA and PP problems are considered separately in the STAPP method, they are all in a single MAS, so the coupling property between them is not neglected.

IV. EXPERIMENT

A. EXPERIMENTAL SETTING

1) ENVIRONMENTAL SETTING

We designed a simulation training environment for MUTAPP based on the OpenAI's platform. It consists of agents, which represent UAVs, targets, and threat areas. The geometric coordinate system is established with the environmental center as the origin of the coordinates and the length of each quadrant in the environment = 1. In this coordinate system, the size of the agent is 0.05, the size of the target is 0.10, and the size of the threat area is 0.15. The locations of threat areas, targets, and agents are randomly generated in each training episode.

There are several indicators used to measure the effectiveness of training, including the collision rate between agent and agent (CRBAA), the collision rate between agent and threat area (CRBAT) and the task completion rate (TCR). Their formulas are shown in (5).

$$\begin{cases} CPBAA = \dfrac{NumberofCollisionsbetweenAgentsandAgents}{Totalmovingstepsofallagents} \\ CPBAT \\ \quad = \dfrac{NumberofCollisionsbetweenAgentsandTheatareas}{Totalmovingstepsofallagents} \\ TCR = \dfrac{Numberoftrainingepisodewithtaskcompletion}{1000trainingepisodes} \end{cases} \tag{5}$$

2) TRAINING PROCESS

Agents can move 50 steps in each training episode. The total number of training episodes is set to 150, 000. In each

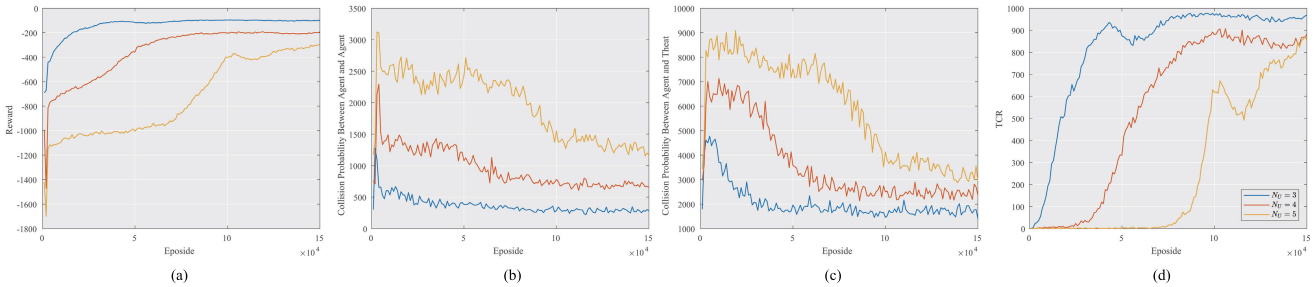


FIGURE 8. Comparison of the reward, CPBAA, CPBAT, and TCR with $N_U = 3, 4, 5$ and $\sigma = 0.1$.

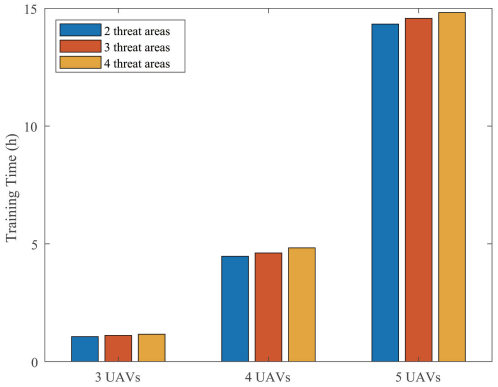


FIGURE 9. Comparison of time for training to convergence with $N_U = 3, 4, 5$.

step, each agent selects an action to be taken only according to its own observation and policy, as per Line 5 of Algorithm 4. And an action is executed according to the formula (6):

$$\vec{a} = \vec{v} * \Delta t + \mathcal{N}_t \tag{6}$$

where the $\vec{a}$ is a displacement vector, the $\vec{v}$ is composited by the Velocity of the x, y axis which are output of the policy, the Δt is a fixed time, and $\mathcal{N}_t$ is a random noise. Once an UAV gets the observation information, it chooses the action according to the above process. When all UAVs complete their actions, the system enters the next state.

One training episode is over when the agents finish the task or 50 steps are taken.

B. EXPERIMENTAL RESULTS

We first conducted a baseline experiment. It used the original MADDPG configuration without enlarging the critical area. As can be seen in Fig. 6, (a) shows that the reward is convergent, while (b) and (c) indicate that the CPBAA and CPBAT decrease greatly after training and also converge. This shows that training is effective and STAPP can be used to solve MUTAPP problems.

Next, we used the method to add critical areas. We set the width $\sigma = 0.1, 0.2, 0.3$ in turn. The experimental results are

TABLE 1. Detailed comparison of results with $\sigma = 0, 0.1, 0.2, 0.3$.

σ	CPBAA	CPBAT	TCR
0(baseline)	0.42%	2.04%	89.95%
0.1	**0.36%**	**1.99%**	**91.01%**
0.2	0.38%	2.08%	90.86%
0.3	0.45%	1.97%	82.87%

shown in Fig. 7 and are compared with the baseline results. The (a) shows that as σ increases, the reward decreases because the collapsible area is expanded. According to (b) and (c), the CPBAA and CPBAT decrease with increases in σ. A more detailed comparison is shown in Table 1. And Table 1 shows that the algorithm achieves the best performance when $\sigma = 0.1$.

In addition, during the initial training period, agents are more likely to collide with each other due to their small range of motion, which leads to the observation that the CPBAA is very large while the CPBAT is not.

We also experimented with $N_U = 4, 5$ ($\sigma = 0.1$) and compared the results with those of the previous experiment, as shown in Fig. 8. As we can see from Fig. 8 (a), as the number of agents increases, the reward becomes smaller and the convergence speed becomes slower and slower. The (b) and (c) show that the CPBAA and CPBAT can also converge. The (d) indicates that as the number of agents increases, it takes longer for agents to learn effective policies.

We compared the training times on a desktop computer (i7-7700 CPU, 8 GB RAM) as Fig. 9 shows. We found that training convergence becomes increasingly difficult as the number of agents increases. Increases in threat areas do not significantly affect training time. This is because the probability of MADDPG exploring the right strategy gradient decreases exponentially with increases in the number of agents.

Fig. 10 shows some behaviors acquired by agents through training. We can see that agents can be assigned to their targets at a relatively close distance through training and there is a clear behavior of avoiding threat areas, just as Fig. 10 (a) and (b) show. However, it is still possible to hit a threat area.

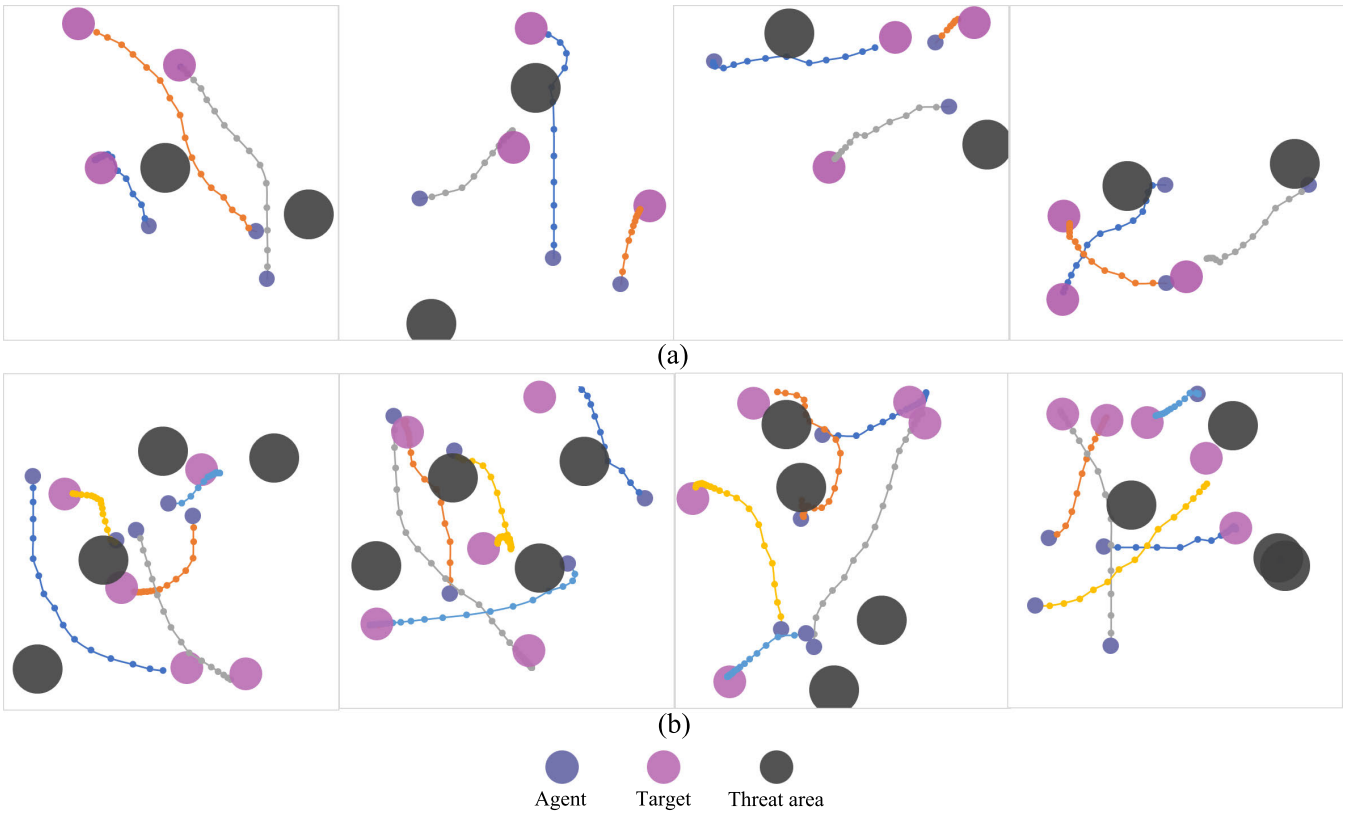


FIGURE 10. **Behaviors of agents in some cases after training:(a) shows 3 UAVs, 3 targets, and 2 threat areas. (b) shows 5 UAVs, 5 targets, and 4 threat areas.**

V. CONCLUSION AND FUTURE WORK

This paper presents an innovative artificial intelligence method named STAPP, which incorporates the most advanced multi-agent reinforcement learning algorithm (MADDPG) to solve MUTAPP problems in dynamic environments. The experiments show that our method is effective. In addition, the STAPP method provides a new way of thinking for researchers in this field. In future, we will work towards training more UAVs and reducing training times as possible as we can. These directions would be explored by more complex networks and grouping UAVs based on mean field theory.

REFERENCES

[1] Q. H. Wang, G. Wan, X. F. Cao, and L. X. Xie, "A modeling method of multiple targets assignment under multiple UAVs' cooperation," *IOP Conf. Ser., Mater. Sci. Eng.*, vol. 187, Mar. 2017, Art. no. 012006. doi: 10.1088/1757-899x/187/1/012006.

[2] R. W. Beard, T. W. McLain, and M. Goodrich, "Coordinated target assignment and intercept for unmanned air vehicles," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 3, May 2002, pp. 2581–2586.

[3] R. Borndörfer, A. Eisenblätter, M. Grötschel, and A. Martin, "Frequency assignment in cellular phone networks," *Ann. Oper. Res.*, vol. 76, pp. 73–93, Jan. 1998.

[4] L. Pardalos and M. Resende, "A greedy randomized adaptive search procedure for the quadratic assignment problem," in *Quadratic Assignment and Related Problems* (DIMACS: Series in Discrete Mathematics and Theoretical Computer Science), vol. 16. Providence, Rhode Island: American Mathematical Society, 1994, pp. 237–261.

[5] C. Schumacher, P. Chandler, and S. Rasmussen, "Task allocation for wide area search munitions via iterative network flow," in *Proc. AIAA Guid., Navigat., Control Conf. Exhibit*, 2002, p. 4586.

[6] P. Chandler and M. Pachter, "Hierarchical control for autonomous teams," in *Proc. AIAA Guid., Navigat., Control Conf. Exhibit*, Aug. 2001, p. 4149.

[7] J. E. Tierno and R. Kreichauf, "Distributed autonomous control of concurrent combat tasks," in *Proc. Eur. Control Conf. (ECC)*, Sep. 2001, pp. 2615–2620.

[8] T. W. McLain, P. R. Chandler, S. Rasmussen, and M. Pachter, "Cooperative control of UAV rendezvous," in *Proc. Amer. Control Conf.*, vol. 3, Jun. 2001, pp. 2309–2314.

[9] R. Kumar and D. C. Hyland, "Control law design using repeated trials," in *Proc. Amer. Control Conf.*, vol. 2, Jun. 2001, pp. 837–842.

[10] L. Singh and J. Fuller, "Trajectory generation for a UAV in urban terrain, using nonlinear MPC," in *Proc. Amer. Control Conf.*, vol. 3, Jun. 2001, pp. 2301–2308.

[11] A. Richards, J. Bellingham, M. Tillerson, and J. How, "Coordination and control of multiple UAVs," in *Proc. AIAA Guid., Navigat., Control Conf. Exhibit*, Aug. 2002, p. 4588.

[12] J. Bellingham, A. Richards, and J. P. How, "Receding horizon control of autonomous aerial vehicles," in *Proc. Amer. Control Conf.*, vol. 5, May 2002, pp. 3741–3746.

[13] L. Lv, S. Zhang, D. Ding, and Y. Wang, "Path planning via an improved DQN-based learning policy," *IEEE Access*, vol. 7, pp. 67319–67330, 2019.

[14] Z. Xiao, B. Zhu, Y. Wang, and P. Miao, "Low-complexity path planning algorithm for unmanned aerial vehicles in complicated scenarios," *IEEE Access*, vol. 6, pp. 57049–57055, 2018.

[15] R. Lowe, Y. Wu, A. Tamar, J. Harb, O. P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6379–6390.

[16] W. C. Stirling and M. A. Goodrich, "Conditional preferences for social systems," in *Proc. IEEE Int. Conf. Syst., Man Cybern. E-Syst. E-Man Cyberspace*, vol. 2, Oct. 2001, pp. 995–1000.

[17] T. Shima, S. J. Rasmussen, and A. G. Sparks, "Uav cooperative multiple task assignments using genetic algorithms," in *Proc. Amer. Control Conf.*, Jun. 2005, pp. 2989–2994.

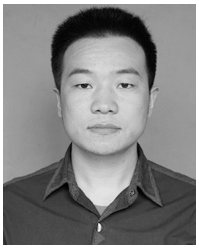
[18] W. Ou, F. Zou, X. Xu, and G. Zheng, "Targets assignment for cooperative multi-UAVs based on chaos optimization algorithm," in *Proc. 9th Int. Conf. Young Comput. Scientists*, Nov. 2008, pp. 2852–2856.
[19] L. Babel, "Coordinated target assignment and UAV path planning with timing constraints," *J. Intell. Robotic Syst.*, vol. 94, nos. 3–4, pp. 857–869, 2018.
[20] J.-L. Liu, Z.-G. Shi, and Y. Zhang, "A new method of UAVs multi-target task assignment," *DEStech Trans. Eng. Technol. Res.*, to be published.
[21] J. Bellingham, M. Tillerson, A. Richards, and J. P. How, "Multi-task allocation and path planning for cooperating uavs," in *Cooperative Control: Models, Applications and Algorithms*. Berlin, Germany: Springer, 2003, pp. 23–41.
[22] M. Alighanbari, "Task assignment algorithms for teams of uavs in dynamic environments," Ph.D. dissertation, Oper. Res. Center, Dept. Aeronaut. Astronaut., Massachusetts Inst. Technol., Cambridge, MA, USA, 2004.
[23] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. M. O. Heess, T. Erez, Y. Tassa, D. Silver, and D. P. Wierstra, "Continuous control with deep reinforcement learning," U.S. Patent 15 217 758, Jan. 26, 2017.
[24] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

HAN QIE received the B.S. degree in computer science from the Harbin Institute of Technology, in 2017. He is currently pursuing the master's degree with the National University of Defense Technology. His research interests include multi-agent reinforcement learning algorithms and UAV formation algorithms.

DIANXI SHI received the B.S., M.S., and Ph.D. degrees in computer science from the National University of Defense Technology, Changsha, China, in 1989, 1996, and 2000, respectively, where he served as an Associate Researcher and the Master Tutor of the Institute of Network and Information Security, College of Computer, from December 2003 to December 2011. From December 2011 to January 2018, he was a Researcher and the Doctoral Supervisor of Parallel and Distributed Processing Laboratory, National Defense Science and Technology. He is currently a Researcher and the Deputy Director of the Artificial Intelligence Research Center, National Innovation Institute of Defense Technology. His research interests include distributed object middleware technology, software component technology, adaptive software technology, and intelligent unmanned cluster system software architecture.

TIANLONG SHEN received the Ph.D. degree in mathematics from the National University of Defense Technology, in 2017. He is currently an Engineer of the Artificial Intelligence Research Center, National Innovation Institute of Defense Technology, Beijing, China. He has published articles on top Mathematics journals and conferences, such as DCDS-B and JMAA. His research interests include mathematics methods with applications in communication networks, and multi-agent reinforcement learning algorithms with applications in real-time strategy games, from 2018.

XINHAI XU received the Ph.D. degree in computer science from the National University of Defense Technology, in 2012. He is currently an Associate Professor with the Artificial Intelligence Research Center, National Innovation Institute of Defense Technology, Beijing, China. His research interests include artificial intelligence algorithm, simulation, and parallel computing.



YUAN LI received the B.S. degree in computer science from the National University of Defense Technology, China, and the Ph.D. degree in network optimization from Lund University, Sweden, in 2008 and 2015, respectively. He is currently an Engineer of the Artificial Intelligence Research Center, National Innovation Institute of Defense Technology, Beijing, China. He has published articles on top network journals and conferences, such as the IEEE JSAC and the IEEE INFOCOM. From 2018, his research interest includes on multi-agent reinforcement learning algorithms with applications in real-time strategy games. His research interests include network modeling, algorithm design, integer programming, and other combinatorial methods with applications in communication networks. He was a recipient of the first place in the Multi-agent Confrontation Competition 2019 which was held by China Electronics Technology Group Corporation.

LIUJING WANG received the B.S. degree in computer science from Shandong University, in 2017. She is currently pursuing the master's degree with the National University of Defense Technology. Her current research interests include computer vision, image process, object detection, and UAV formation algorithms.