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

HAN QIE¹, DIANXI SHI²,³, TIANLONG SHEN², XINHAI XU², YUAN LI², LIUJING WANG¹

¹College of Computer, National University of Defense Technology, Changsha, Hunan 410073, China (e-mail: qiehan@foxmail.com)
²Artificial Intelligence Research Center, National Innovation Institute of Defense Technology, Beijing 100071, China
³Tianjin Artificial Intelligence Innovation Center, Tianjin 300457, China

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

This work was supported by the National Key Research and Development Program of China(2017YFB1001901), the National Defense Science and Technology Foundation for Young Scientists of China (No. 030403), and the National Natural Science Foundation of China (No. 11801563, 91648204, 61532007 and 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

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPBAA</td>
<td>Collision Probability Between Agent and Agent</td>
</tr>
<tr>
<td>CPBAT</td>
<td>Collision Probability Between Agent and Threat</td>
</tr>
<tr>
<td>CWS</td>
<td>Cell Winnings Suppression</td>
</tr>
<tr>
<td>DDPG</td>
<td>Deep Deterministic Policy Gradient</td>
</tr>
<tr>
<td>DQN</td>
<td>Deep Q-learning Network</td>
</tr>
<tr>
<td>GAs</td>
<td>Genetic Algorithms</td>
</tr>
<tr>
<td>MADDPG</td>
<td>Multi-Agent Deep Deterministic Policy Gradient Algorithm</td>
</tr>
<tr>
<td>MARL</td>
<td>Multi-Agent Reinforcement Learning</td>
</tr>
<tr>
<td>MAS</td>
<td>Multi-Agent System</td>
</tr>
<tr>
<td>MDPs</td>
<td>Markov Decision Processes</td>
</tr>
<tr>
<td>MILP</td>
<td>Mixed-Integer Linear Programming</td>
</tr>
<tr>
<td>MLP</td>
<td>Multi-Layer Perception</td>
</tr>
<tr>
<td>MUTAPP</td>
<td>Multi-UAV Target Assignment and Path Planning</td>
</tr>
<tr>
<td>PP</td>
<td>Path Planning</td>
</tr>
<tr>
<td>PSO</td>
<td>Particle Swarm Optimization</td>
</tr>
<tr>
<td>RHTA</td>
<td>Receding Horizon Task Assignment</td>
</tr>
<tr>
<td>SDT</td>
<td>Satisfaction Decision Theory</td>
</tr>
<tr>
<td>STAPP</td>
<td>Simultaneous Target Assignment and Path Planning</td>
</tr>
</tbody>
</table>

I. INTRODUCTION

Recent developments in unmanned aerial vehicles (UAVs) have made them indispensable tools in modern society. They 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, 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 preassigned altitude, whereby 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].

FIGURE 1. Multi-UAV Targets Assignment Scenario

FIGURE 2. Execution process is regarded as MDPs
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 \( \pi \) 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 \( U_i \in U_i, i = 1, \ldots, N_U \) in a two-dimensional environment. The position of UAV \( U_i \) at time \( t \) is given by \((x_i^U(t), y_i^U(t))\). The targets \( T_i \in T, i = 1, \ldots, N_T \) and threat areas \( D_i \in D, i = 1, \ldots, N_D \) are distributed across it. The position of target \( T_i \) is denoted by \((x_i^T, y_i^T)\), while threat \( D_i \) is given by \((x_i^D, y_i^D)\). For UAV \( U_i \), a path \( \{P_i\} \) is planned and given by (1) and the length of \( \{P_i\} \) is given by \( d_i \).

\[
P_i = \{(x_i^U(0), y_i^U(0)), (x_i^U(1), y_i^U(1)), \ldots, (x_i^U(n), y_i^U(n))\}
\]

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

1. Each target \( T_i \) is assigned to one UAV, i.e., \( \bigcup_{i=1}^{N_U} T_i = T, i \in \{1, \ldots, N_T\} \).
2. Each target is only assigned to one UAV, i.e., \( \forall i \neq j, T_i \neq T_j \) for \( i, j \in \{1, \ldots, N_T\} \).
3. Each UAV’s path is collision-free with threat area, i.e., \( \forall i, j, D_j \notin P_i \) for \( i \in \{1, \ldots, N_U\}, j \in \{1, \ldots, N_D\} \).
4. Each UAV’s path is collision-free with other UAVs, i.e., \( \forall i, j, (x_i^U(t), y_i^U(t)) \neq (x_j^U(t), y_j^U(t)) \) 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 \( \theta = \{\theta_1, \theta_2, \ldots, \theta_{N_U}\} \), and let \( \mu = \{\mu_{\theta_1}, \mu_{\theta_2}, \ldots, \mu_{\theta_{N_U}}\} \) (abbreviated as \( \mu_i \)) be the set of all agent deterministic policies. For the deterministic policy \( \mu_i \) of agent \( i \), the gradient can be written as (2).

\[
\nabla_{\theta_i} J(\mu) = E_{a,a'\sim D} \left[ \nabla_{\theta_i} \mu_i(a|o) \nabla_{a'} Q_{\mu_i}^T(x,a_1,\ldots,a_N)|a_i=\mu_i(o_i) \right]
\]

where \( x = \{o_1, o_2, \ldots, o_{N_U}\} \) represent the state, \( Q_{\mu_i}^T(x,a_1,\ldots,a_{N_U})|a_i=\mu_i(o_i) \) is the \( Q \)-value function, \( o_i \) is the action of agent \( i \) and \( a_i \) 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 \( D \) represents the experience replay [24] buffer which contains a series of tuples \( (x, x', a_1, \ldots, a_{N_U}, r_1, \ldots, r_{N_U}) \), recording the experiences of all agents. The \( x' \) is the new state of the MAS after executing the actions and the \( r_i \) is the
FIGURE 3. The framework of MADDPG

The reward of agent $i$. The critic-network $Q^i_\mu$ is updated by the loss function written as (3).

$$L(\theta_i) = \mathbb{E}_{x,a,r,x'}[(Q^i_\mu(x, a_1, \ldots, a_{N_U}) - y)^2]$$

where $y = r_i + \gamma Q^i_\mu(x', a'_1, \ldots, a'_{N_U}) | a'_j = \mu'_j(o_j)$

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

$$\nabla_\theta J \approx \frac{1}{S} \sum_j \nabla_\theta \mu_i(o_j) \nabla_a Q^i_\mu(x^j, a^j_1, \ldots, a^j_{N_U}) | a^j_i = \mu_i(o^j_i)$$

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 sub-problems 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 $r_1$

1: Initialize $r_1 = 0$
2: for target 0 to $N_T$ do
3: Calculate the distance set $d$ of all agents to the target
4: $r_1 = r_1 + (\min(d))$
5: end for
6: 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 indicators. The reward structure used to train the UAV collision avoidance is expressed as Algorithm 2.

**Algorithm 2 Calculate reward r_2**

1. Initialize r_2 = 0
2. for agent 0 to N_U do
3. if dist_2 < dist_{min_2} + σ then
4. r_2 = r_2 - (dist_{min_2} + σ - dist_2)
5. else
6. r_2 = r_2 - 0
7. end if
8. end for
9. return r_2

where dist_{min_2} = 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, where dist_{min_3} = 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.

**Algorithm 3 Calculate reward r_3**

1. Initialize r_3 = 0
2. for agent 0 to N_U do
3. if dist_3 < dist_{min_3} + σ then
4. r_3 = r_3 - (dist_{min_3} + σ - dist_3)
5. else
6. r_3 = r_3 - 0
7. end if
8. end for
9. return r_3

**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 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_t represents the observation of agent i and 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.

**Algorithm 4 STAPP**

1. Initialize N_U, N_T, N_D, and σ
2. for episode = 1 to M do
3. Receive initial state x
4. for t = 1 to MAX-Step do
5. for each agent i, select action a_i = \mu_{\theta_i}(o_i) + \mathcal{N}_t w.r.t. the current policy and observation
6. Execute actions and observe reward r = r_1 + r_2 + r_3 and new state x'
7. Store experience in replay buffer \mathcal{D}
8. for each agent i = 1 to N_U do
9. Sample a random minibatch of S samples from \mathcal{D}
10. Update the Critic network by minimizing the loss as formula (3)
11. Update the Actor network using the sampled policy gradient as formula (4)
12. end for
13. Update the target network for each agent i
14. end for
15. end for

**IV. EXPERIMENT**

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

<table>
<thead>
<tr>
<th>$\sigma$</th>
<th>CPBAA</th>
<th>CPBAT</th>
<th>TCR</th>
</tr>
</thead>
<tbody>
<tr>
<td>0(baseline)</td>
<td>0.42%</td>
<td>2.04%</td>
<td>89.95%</td>
</tr>
<tr>
<td>0.1</td>
<td>0.36%</td>
<td>1.99%</td>
<td>91.01%</td>
</tr>
<tr>
<td>0.2</td>
<td>0.38%</td>
<td>2.08%</td>
<td>90.86%</td>
</tr>
<tr>
<td>0.3</td>
<td>0.45%</td>
<td>1.97%</td>
<td>82.87%</td>
</tr>
</tbody>
</table>

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.
V. CONCLUSIONS 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


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


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.

XINHAI XU received his PhD degree in computer science from the National University of Defense Technology in 2012. He is now an Associate Professor of Artificial Intelligence Research Center, National Innovation Institute of Defense Technology, Beijing, China. His research interest lies in artificial intelligence algorithm, simulation and parallel computing.

YUAN LI received his BSc degree in computer science at National University of Defense Technology, China, and PhD degree in network optimization at Lund University, Sweden, in 2008 and 2015, respectively. His research interests are in network modeling, algorithm design, integer programming and other combinatorial methods with applications in communication networks. He has published papers on top network journals and conferences, such as IEEE JSAC and IEEE INFOCOM. From 2018, his research interest is focused on multi-agent reinforcement learning algorithms with applications in real-time strategy games. He has won 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, in 2017, from Shandong University. She is currently working toward the master’s degree in the National University of Defense Technology. Her current research interests include computer vision, image process, Object Detection and UAV formation algorithms.