2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Dynamic Multi-Agent Deep Deterministic Policy Gradient for Autonomous Navigation of Reconfigurable Unmanned Aerial Vehicle

Lu Xin, Wu Zegui, Zhao Ruqing, Li Fusheng*

Abstract— **The reconfigurable unmanned aerial vehicle (R-UAV) has the ability to create and break physical links to self-assemble and self-disassemble in midair. For the changes in task or environment, this system can dynamically disassemble the rectangular structure into multiple individual UAV modules or integrate these UAV modules into a whole. For practical applications, the R-UAV requires collaborative decision-making for autonomous navigation in complex environments. However, the navigation problem of the R-UAV has not been investigated. In this paper, we propose a dynamic multi-agent deep deterministic policy gradient (DMADDPG) algorithm for autonomous navigation of R-UAV. This algorithm introduces the leader agent assignment mechanism and a collaborative experience reward. The former deals with the action conflict problem caused by the disappearance of the UAV agent when multiple UAV modules are assembled. The latter provides guidance for the UAV agent to plan a collision-free and efficient trajectory. We validate our strategy in both simulation and practical scenarios, and experimental results demonstrate that the proposed scheme can generate reasonable and efficient paths for R-UAV in the presence of obstacles. The experiment video is available at https://youtu.be/mVm0qCvB7HY.**

I. INTRODUCTION

Due to recent advances, unmanned aerial vehicles (UAVs) have been widely applied to various areas including disaster rescue [1], surveillance and monitoring [2], and crop protection [3]. However, in complex application scenarios, a single UAV or cluster of UAVs is difficult to adapt to changes in the working environment due to the limitations of their structure. To address this issue, the reconfigurable unmanned aerial vehicle (R-UAV) has gradually attracted the attention of researchers, which can be broken down into multiple small modules or combined into a whole. The R-UAV provides a feasible solution to complete the task in large-scale complex scenarios. This system consists of multiple UAV modules. In the case of cluttered environments, the UAV modules are able to fly as individual modules for rapid movement. When in a proper place, they cooperate with each other to assemble a temporary structure capable of carrying more weight.

The earliest R-UAV is a distributed flight array provided in [4], which is capable of docking with its peers and flying in a coordinated fashion. However, its individual aerial modules cannot fly independently. In 2015, The "LIFT!" [5] research project launched by Boeing company proposed a modular multi-rotor UAV cluster, which can be manually reconfigured based on payload requirements. However, this operation only increases the payload capacity. In 2018, the ModQuad developed by David et al. [6] was the first flying modular robot to truly realize inflight self-assembling. Furthermore, Guanrui et al [7] proposed ModQuad-Vi unmanned aerial system, which introduced visual perception. The vision-based method can accurately align the docking structure to complete the aerial assembly of the UAV modules. However, both ModQuad and ModQuad-Vi can complete self-assembly and cannot perform disassembly. In 2019, David et al. [8] again proposed an R-UAV, which can disassemble the structure into a single module in the air with sequential separation action, but the sequential splitting scheme is slow and introduces violent oscillation. This R-UAV can complete the self-disassembly, but cannot complete the self-assembly well. In 2021, Diego et al. [9] proposed H-ModQuad, which increased both the payload capacity and the controllable degrees of freedom from 4 to 5 and 6 by connecting different types of modules. However, H-ModQuad is a manually reconfigurable drone that cannot autonomously undergo reconfiguration in complex tasks. Nevertheless, the R-UAV has significant application potential due to their flexibility.

For this application, it is essential to develop a technique to make decisions that allow each UAV module to navigate in complex environments. Recently, a multitude of methods has been proposed for the problem of autonomous multi-UAV navigation. Among them, reinforcement learning-based algorithms have attracted substantial research interest. Soyi et al. [10] proposed a distributed MARL algorithm for controlling and managing the logistics transportation of UAV clusters. It can calculate the optimal transportation route considering UAV cluster collisions, multi-target transportation behaviors, and the battery status of each UAV. Han et al. [11] proposed a path-planning method for UAV clusters based on the multi-agent deep deterministic policy gradient (MADDPG). This method can effectively address the decision-making problem of UAV cluster path planning in dynamic environments with good real-time performance. To achieve autonomous navigation, the work [12] investigated the potential of multi-agent reinforcement learning (MARL) to coordinate multiple UAVs for object tracking. Following the idea MARL, researchers address navigation of multi-UAV by developing a two-stage MARL [13], constrained MARL [14], cooperative MARL [15], and so on.

MARL can be roughly divided into three architectures, decentralized control, centralized control, and centralized training distributed execution [16]. On the issue of cooperative decision-making of the R-UAV system, using a decentralized control architecture cannot coordinate the various

This work was supported by the National Natural Science Foundation of China (No. 62075028)

The authors with School of Automation Engineering, University of Electronic Science and Technology of China, China lifusheng@uestc.edu.cn

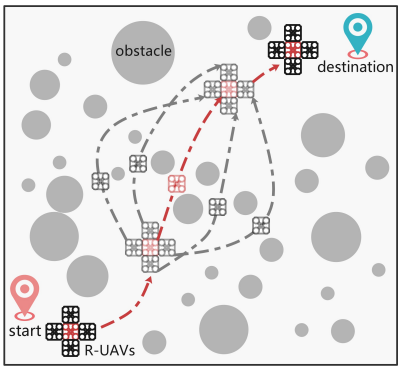

Fig. 1. Trajectory of R-UAV reaching destination in complex environment with obstacles.

UAV modules well. On the other hand, using a centralized control architecture places high demands on the speed and stability of the communication system. The architecture of centralized training distributed execution allows agents to train with a central controller and interact independently with the environment when deployed. Therefore, this paper adopts a centralized training distributed execution architecture. Based on this framework, a series of typical MARL algorithms have been developed, including VDN [17], MADDPG [18], QMIX [19], and others. In multi-agent systems, the MADDPG algorithm can effectively handle the complex cooperation and competition among multiple agents. However, for the R-UAV system, when multiple UAV modules are assembled, there will be a situation in which multiple agents control one combination in the MADDPG algorithm. In this case, different neural networks deployed in corresponding agents do not output exactly equal decisions, leading to control conflict issues. Therefore it is not applicable. In this paper, we will address this problem by creating a novel algorithm that imbues leader agent assignment mechanism and collaborative experience reward into MADDPG.

Fig. 1 shows the autonomous navigation of a scenario where R-UAV collaboratively traverses through obstacles. The objective of this system is to keep the combination flying in a spacious environment. Usually, R-UAV with larger power configurations have larger payloads and longer endurance. When encountering a narrow environment (with obstacles) and the configuration cannot fly through, the combination will be disassembled to pass through.

To the best of our knowledge, the autonomous navigation research of the R-UAV with self-disassembly and self-assembly properties has not yet been studied. Thus, the proposed approach will guide the autonomous navigation of R-UAV in case of complex environments in the future. The contribution of this paper can be concluded as follows:

1) This paper proposes a novel dynamic multi-agent deep deterministic policy gradient (DMADDPG) method for autonomous navigation of R-UAV. This scheme introduces a leader agent assignment mechanism to tackle the issue of decision conflicts when multiple UAV agents are assembled to form one agent. In addition, a collaborative experience reward is developed to collaboratively process simultaneous demands.

2) This paper conducts simulation and practical experiments based on self-developed R-UAV. The results demonstrate that the proposed algorithm can be effectively applied to R-UAV to realize automatic navigation.

II. MADDPG FOR R-UAV SYSTEM

Consider a R-UAV system with N UAV agents parameterized by $\theta = \{\theta_1, \ldots, \theta_N\}$. The cumulative expected reward of the agent i is defined as

$$J(\theta_i) = E[\sum_{t=0}^{\infty} \gamma^t r_t^i] \tag{1}$$

where r is the reward, γ is the attenuation coefficient and t denotes time. For deterministic policy μ_{θ_i} of i-th agent a_i, the gradient of $J(\theta_i)$ can be written as

$$\begin{aligned} &\nabla_{\theta_i} J(\mu_i) \\ &= E_{x,a\sim D}[\nabla_{\theta_i}\mu_i(a_i|o_i)\nabla_{a_i} Q_i^{\mu}(x, a_1, \ldots, a_N)|_{a_i=\mu_i(o_i)}] \end{aligned} \tag{2}$$

where x is the state and o denotes the observation. D is the the experience replay buffer which stores $(x, x', a_1, \ldots, a_N, r_1, \ldots, r_N)$. x' is the new observation state after the action is executed. Q_i^{μ} is the critic network, and μ_i is the actor network [18].

MADDPG uses fixed network computing technology. Each agent has four networks, including actor network, critic network, actor target network and critic target network. During training, the input of the actor network is the observation state o_i of the i-th UAV agent, and the corresponding action $a_i = \mu_{\theta_i}(o_i)$ is output. The input of the critic network is the observation state and actions of all UAV agents, and the output is the evaluation value. After all UAV agents perform corresponding actions, they obtain a new observation state x' and reward r. Then the experience $(x, x', a_1, \ldots, a_N, r_1, \ldots, r_N)$ is stored in the experience replay buffer D, and the current observation state x is updated to the new observation state x'. Then continue to interact with the environment until the experience replay buffer reaches a certain threshold. Randomly sample from the experience replay buffer, and update the critic network Q_i^{μ} of each UAV agent by minimizing the following loss function:

$$L(\theta_i) = \frac{1}{S}\sum_j \left(y^j - Q_i^{\mu}(x, a_1, \ldots, a_N)\right)^2 \tag{3}$$

where S is a random minibatch size of samples and $y^j = r_i^j + \lambda Q_i^{\mu'}(x'^j, a_1', \ldots, a_N')|_{a_k'=\mu_k'(o_k^j)}$.

Update the actor network for each UAV agent using the following policy gradient:

$$\nabla_{\theta_i} J \approx \frac{1}{S}\sum_j \nabla_{\theta_i}\mu_i(o_i^j)\nabla_{a_i} Q_i^{\mu}(x^j, a_1^j, \ldots, a_N^j)|_{a_i=\mu_i(o_i)} \tag{4}$$

After a certain number of trainings, each target network is soft updated by the following formula:

$$\theta_i' \leftarrow \tau\theta_i + (1-\tau)\theta_i' \tag{5}$$

where τ is the weight parameter.

During execution, the critic network is removed and the actor network is deployed on the corresponding UAV side. This process shows that the adopted MADDPG is centralized training and decentralized execution.

Limitations of MADDPG applied in the R-UAV: The first is the agent disappearance problem. For our R-UAV, during the self-assembly process, multiple UAV agents in the MADDPG framework become one UAV agent at the moment of assembly. Unfortunately, this issue has not been considered in MADDPG. This is because applying MADDPG to the R-UAV system leads to the special case of multiple agents controlling a single rigid body. The second is how to design a suitable reward function to achieve multiple optimization objectives. Under the R-UAV system, in order to safely and quickly traverse complex environments, both reconstruction position assignment (RPA) and path planning (PP) issues need to be considered. That is, RPA requires only one auxiliary agent to choose for each reconstruction position, and the sum of the cruising paths of all auxiliary agents is the shortest. PP requires planning the shortest path to the selected reconstruction location and avoiding collisions.

III. DMADDPG for Autonomous Navigation of R-UAV

A. Problem formulation

Suppose that N cooperative agents are denoted as $U = \{u_1, u_2, \ldots, u_N\}$. The position of u_i at time t is defined as (x_i, y_i, h_i). The obstacle O_j with different height is randomly generated at position $(a_j, b_j, 0)$. The position of destination is fixed as $(x, y, 0)$. The trajectory T_i of the u_i is given by $T_i = \{(x_i(0), y_i(0), h_i(0)), (x_i(1), y_i(1), h_i(1)), \ldots, (x, y, 0)\}$. The length of T_i is defined as d_i

For this scenario, the optimization goal of the system is $\min \sum_{i=1}^{N} d_i$ and trying to keep the combined configuration for maximum weight load. The system needs to consider the following issues.

1) Each agent's trajectory should be collision-free. In the event that this combined UAV platform cannot traverse a complex obstacle environment, it can be flown through alone by breaking up into multiple UAV modules in midair.

2) UAV module agents must not collide with each other while flying as independent modules.

3) When distributed UAV agents need to be assembled, only one UAV agent can be connected to one reconstruction location.

B. DMADDPG

We describe in Figure 2 how to apply DMADDPG to the R-UAV controller. The DMADDPG decision making module outputs the expected angle θ^* and expected position velocity V^* of the UAV. Transmit expected position velocity and feedback position velocity to the position velocity controller. Transmit the expected angle value and feedback angle value to the angle controller. The control distributor

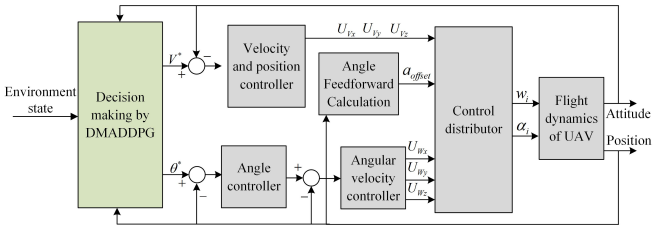

Fig. 2. DMADDPG applied to R-UAV controller.

outputs the control values of all motors and servos. The proposed DMADDPG is based on the MADDPG algorithm and designed with two improved mechanisms including the leader agent decision assignment mechanism and the cooperative experience reward.

1) leader decision assignment mechanism: As shown in Fig. 3, the red module is used to load cargo, the others are auxiliary modules. We define the load module as the leader agent and others as auxiliary agents. When the combined platform encounters an obstacle that cannot be passed, the auxiliary agents need to detach from the combination. The position-velocity vector of the detached auxiliary agent will be different from that of the leader agent. When no complex obstacles are encountered, R-UAV performs the mission in a combined form. At this time the position-velocity vector of the other auxiliary agents should be identical to that of the leader agent. Therefore, each UAV module controller needs to input exactly the same value, otherwise the controller will generate an error. However, it is impossible for actor networks deployed in different UAV agents to output completely equal decisions.

To handle this, we add a "multi-way switch" between the agent and the UAV's distributed controller. When each auxiliary agent is in the same combination with the leader agent, and the action decisions of each auxiliary agent are similar to those of the leader, it is considered that the auxiliary agent obeys the leader agent. Therefore, the input of distributed controllers of each module is uniformly determined by the leader agent, and no longer receives the action decisions of its corresponding auxiliary agent. When the action decision of the auxiliary agent is quite different from that of the leader agent (in case of complex scenes, it needs to leave the combination), the module controller no longer receives the action decision of the leader agent, but directly receives the action decision of the corresponding auxiliary agent. Using this mechanism can ensure that the inputs of the distributed controllers in the combination are exactly the same without causing oscillation of the control system.

This mechanism can be explained in Fig. 3. Assume that the actor network output action vector of the leader agent is $a_{lead} = [vx_{lead}, vy_{lead}]$ and that of $i-th$ auxiliary agent is denoted as $a_i = [vx_i, vy_i]$. When the difference between a_i and a_{lead} is small, each controller receives the decision of the leader agent, otherwise, the controller accepts the decision of its corresponding agent. Therefore, the leader mechanism

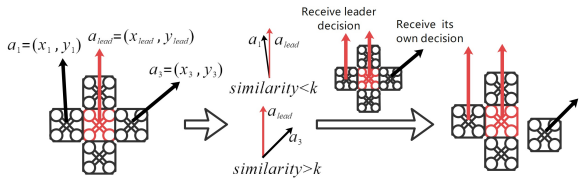

Fig. 3. The process of receiving decision-making for each module of R-UAV.

needs a metric to quantify the difference of action vectors. In this paper, we consider that two action vectors with the same direction can be considered similar in decision making. Therefore, the cosine similarity is used as the measure, which is given by

$$\begin{aligned} similarity &= \frac{a_i \times a_{lead}}{\|a_i\| \|a_{lead}\|} \\ &= \frac{vx_i \times vx_{lead} \times vy_i \times vy_{lead}}{\sqrt{vx_i^2+vy_i^2}+\sqrt{vx_{lead}^2+vy_{lead}^2}} \end{aligned} \tag{6}$$

When $similarity > k$, $k \in [-1, 1]$ each controller receives the decision output by the leader agent, otherwise, each controller receives the decision output by its corresponding agent.

2) Collaborative experience reward: The collaborative experience reward consists of auxiliary agent reward and leader agent reward.

Auxiliary agent reward: To define the reward for the auxiliary agent, we consider two issues, namely RPA and the PP. For RPA, the auxiliary agent needs to determine at which position of the leader agent to combine and the sum of the cruise paths of all auxiliary agents is the shortest. For PP, it is necessary to plan the shortest path to the destination and avoid collisions.

For RPA, we first compute the set of distances D_{au} of M auxiliary agents to the $i-th$ reconstruction position p_i, where $D_{au} = \{d_{au}^1, d_{au}^2, \ldots, d_{au}^M\}$. Define the shortest distance from the reconstruction position p_i to the auxiliary agent as dr_{p_i} where $dr_{p_i} = \min(D_{au})$. Thus, the reward function for RPA is defined as $dr = \sum_i dr_{p_i}$.

For PP, in order to avoid collisions between auxiliary agents, the pseudo-collision mechanism is introduced. Since the auxiliary agent only reassembles around the leader agent, the pseudo-collision mechanism is canceled when it is around the leader agent. Suppose the radius of the warning area is σ, and the radius of the auxiliary agent is r_u. The distance between two auxiliary agents is defined as $dist^{u\&u}$. If $dist^{u\&u} < 2 \times r_u + \sigma$, it has $rb = 2 \times r_u + \sigma - dist^{u\&u}$, else $rb = 0$. Therefore, the first reward term of PP can be obtained by $pr = \sum_j rb_j$.

Another issue of PP is handling collisions between auxiliary agents and obstacles. Define the radius of the obstacle as r_o, and the distance between each agent and the obstacle as $dist^{u\&o}$. Similarly, another reward term for PP is $or = \sum_j rc_j$ where $rc = r_o + r_u + \sigma - dist_i^{u\&o}$.

Consider dr, pr, and or comprehensively, the reward of the auxiliary agent is given by

$$r_{au} = -(\lambda_1 dr + \lambda_2 pr + \lambda_3 or) \tag{7}$$

It is worth noting that the reward dr conflicts with the reward or when encountering an obstacle. In this case, the reward dr forces the auxiliary agent to stay in the reconstruction position to fly in a combination to maximize the reward. However, the reward or will force the auxiliary agent away from the obstacle to reduce the collision penalty. For this reason, when the coefficient λ_1 is too large, the auxiliary agent will ignore the obstacles and continue to obey the leader to maintain the configuration flight, resulting in a collision and crash. When the coefficient λ_3 is too large, the agent will easily disobey the leader and leave the combination, even if each module can pass through obstacles without leaving the combination. At this time, it will be difficult for the combination to maintain a certain configuration for a long time to complete the flight, resulting in a decline in the battery life of the load module.

Leader agent reward: For the leader agent, its goal is to go to the destination and avoid collisions, which is a PP problem. The reward consists of two parts. The first is given by $dr = dist_l$ where $dist_l$ represents the distance between the leader agent's current position to the destination.

The second is to deal with the problem of the leader agent colliding with obstacles. The reward or is introduced. The obstacle radius is given by r_o, the warning area radius is σ, and the leader UAV radius is r_l. The d_{l-o} is calculated by the distance between the leader UAV and the obstacle. If $d_{l-o} < r_o + r_l + \sigma$, it has $p = r_o + r_l + \sigma - d_{l-o}$. Otherwise, $p = 0$. Therefore, for all obstacles, it can be obtained $or = \sum_f p_f$.

Finally, the two parts of rewards are superimposed with different weights, and the reward of the leader UAV agent is obtained as follows:

$$r_{lead} = -(\lambda_1 dr + \lambda_3 or) \tag{8}$$

3) Formulation of DMADDPG: For the formulation of DMADDPG, the crucial parts including state space, action space, and network design should be defined.

State space: The state space of our DMADDPG consists of velocity information and position information. Assume Av_{xi} and Av_{yi} respectively represent the velocity in the x and y directions of the $i-th$ auxiliary agent, Ax_i and Ay_i represent the x, y coordinates of the $i-th$ auxiliary agent, Bx_j and By_j represent the x, y coordinates of $j-th$ obstacle, Rx_f and Ry_f represent x, y coordinates of $f-th$ reconstruction position. Then, the observation state of the $i-th$ auxiliary agent can be defined as $o_i = [Av_{xi}, Av_{yi}, A_{1i}, \ldots, A_{Ni}, B_{1i}, \ldots, B_{ki}, R_{1i}, \ldots, R_{Ni}]$ where $A_{1i} = [Ax_i - Ax_1, Ay_i - Ay_1, Av_{x1}, Av_{y1}]$, $B_{1i} = [Ax_i - Bx_1, Ay_i - By_1]$, $R_{1i} = [Ax_i - Rx_1, Ay_i - Ry_1]$. Similarly, the observation state of the leader agent can be defined as $o_i = [Av_{xi}, Av_{yi}, A_{1i}, \ldots, A_{Ni}, B_{1i}, \ldots, B_{ki}, T]$ where $T = [Ax_i - Tx, Ay_i - Ty]$ means difference between

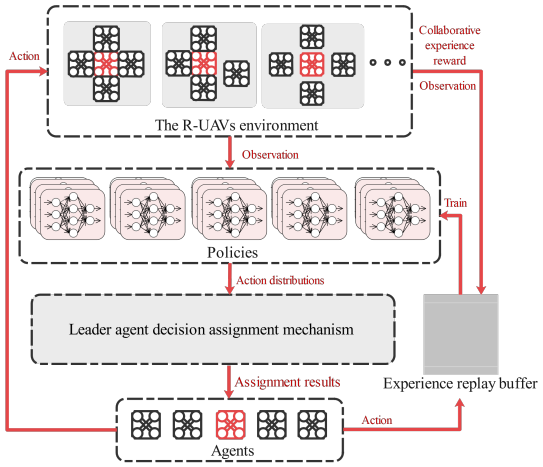

Fig. 4. The workflow of DMADDPG.

current leader agent coordinate (Ax_i, Ay_i) and destination coordinate (Tx, Ty).

Action space: The main actions of each UAV include controlling the direction of motion and the on-off control of the UAV's magnet during reassembly. In this design, when the auxiliary agent reaches the reconstruction position and obeys the leader's decision, then the magnet is automatically controlled to be turned on. When the auxiliary agent is at the reconstruction position, but the leader agent judges that this auxiliary agent does not obey the leadership, the magnet is automatically controlled to be disconnected.

Therefore, the on-off control of the magnet can be completed autonomously, without the need for reinforcement learning to directly output actions for decision-making. For this reason, we define the action vector of the $i-th$ agent in DMADDPG as $V_i = [V_{a_i}^x, V_{a_i}^y]$, where $V_{a_i}^x$ and $V_{a_i}^y$ represent the speed in the x and y directions, respectively.

Network design: Both the actor network and the critic network use a fully-connected neural network with 4 hidden layers. The number of units in each layer is 256, 512, 256, and 64, respectively, and the ReLU [20] activation function is used. The number of input layer units of the actor network is determined by the state space dimension, and the number of input layer units of the critic network is determined by the state space, action space, and the number of agents. The output layer of the actor network and the critic network does not use an activation function.

The workflow of DMADDPG is shown in Fig. 4.

IV. Performance Evaluation

A. Simulation setup

The simulation environment for DMADDPG consists of five UAV agents, a destination, and a high density of obstacles. The coordinate system is established and normalized with the center of the environment as the coordinate origin. The length of each quadrant in the environment is 1. In this coordinate system, the radius of the UAV module is 0.05 and that of the obstacle is 0.05. Each training set randomly

TABLE I

Experimental hyper-parameter settings

hyper-parameter	Value
Network optimizer	Adam
Learning rate	0.01
Discount factor	0.97
Experience replay buffer size	10^6
Batch size	1256

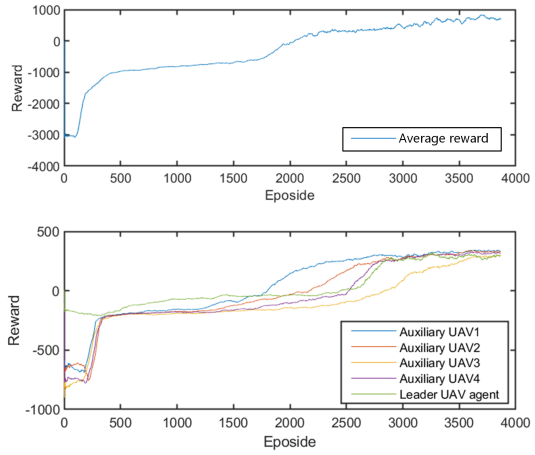

Fig. 5. The reward convergence curve.

generates the coordinates of obstacles, the coordinates of the destination, and the initial coordinates of each UAV. The settings of some hyper-parameters in the algorithm training process are shown in Table 1.

B. Simulation results

We first conducted a preliminary experiment to investigate the reward convergence. Here, five UAV agents are selected to build the system of R-UAV and each agent can move 250 steps in each training episode. The total number of training episodes is 40000. Through several practices we determined that $\lambda_1 = 0.1$, $\lambda_2 = 0.4$, $\lambda_3 = 0.7$. The reward convergence curve is shown in Fig. 5.

As shown in Fig. 5, the upper one is the average reward of all UAV agents which converges after 30000 training episodes. The below one of Fig. 5 shows the reward for each agent. The leader agent's reward increases rapidly in the early training phase. This is because the lead agent only needs to go directly to the destination. On the contrary, auxiliary agents need to cooperate in choosing the reconstruction position in the early stage. In the later stage of training, each auxiliary agent can quickly reach its own reconstruction position, and the leader agent needs to ensure that the combination does not collide, so the convergence speed is slow.

Fig. 6 shows the action of the five agents as they traverse the complex obstacle environment in a like "+" configuration. The red line denotes the movement trajectory of the leader agent. The initial position of each UAV module is randomly generated (Fig. 6(a)). In order to achieve the maximum load, the system tries to fly in a combined form (Fig. 6(b)). When

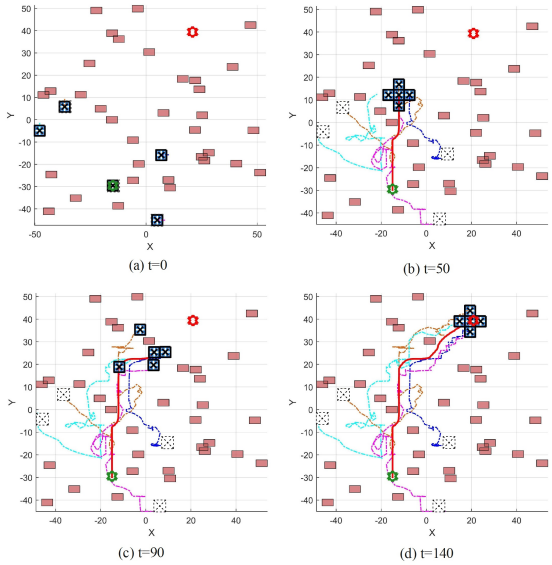

Fig. 6. (a) t=0, the initial status of each UAV module; (b) t=50, five UAV modules complete the combination; (c) t=90, unable to traverse narrow environment, the combination disassembles; (d) t=140, five UAV modules reassembles to maintain configuration

the combined platform encounters an obstacle, the auxiliary agents will disengage (Fig. 6(c)). After traversing the obstacles, each auxiliary agent will go to the reconstruction position of the lead agent to assemble (Fig. 6(d)).

C. Practical experiment setup

In the practical experiment, we use the optical infrared motion capture system to observe the state of the environment. The motion capture system uses a switch to transmit the data of the four infrared cameras to the PC for calculation, and can accurately obtain the position information and attitude information of all UAV modules and obstacles in the field. The PC side sends the environmental status and mission objectives to each UAV module through the router. The experimental platform includes the ZVR optical camera, solving system, switch, and router. The calculating frequency of the motion capture system is 200Hz.

The computing platform of the UAV module is composed of ZYNQ7020 and STM32F407. ZYNQ7020 deploys the actor network of DMADDPG and connects to the wifi module to receive the position and attitude data of the R-UAV and obstacles sent by the router through the UDP protocol. Based on the information received, the actor network generates action decisions in real-time. STM32F407 deploys a distributed control algorithm, receives the action decision of ZYNQ7020, and controls the R-UAV.

Here, the R-UAV consists of four UAV modules, as shown in Fig. 7. The connection structure consists of a copper solenoid sucker, which is connected to the relay. The flight control system can control the physical link by controlling the on and off of the electromagnetic coil. The UAV module uses a 12g digital steering gear to control the inclination

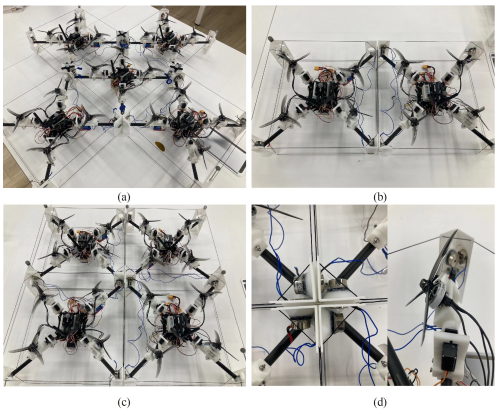

Fig. 7. The R-UAV combined with (a) 5 UAV modules, (b) 4 UAV modules and (c) 2 UAV modules. (d) The connection structure and tilting structure.

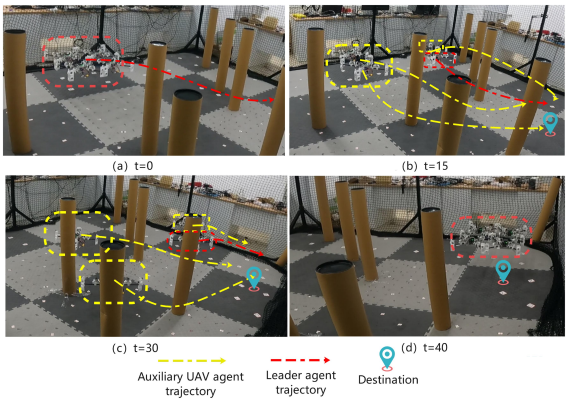

Fig. 8. (a) t=0, the state of the combination at the initial moment; (b) t=15, the combination cannot pass through the narrow environment, so it needs to be disassembled; (c) each UAV module traverses a dense obstacle environment; (d) All UAV modules reassembled.

angle of the rotor to realize vector control. The main body of the UAV module uses 3D printing and carbon solder, and the power system uses a 4000mah lithium battery and a KV1950 motor. There are four UAV modules and seven cylindrical obstacles in the experimental scene. The decision-making and reconstruction process is shown in Fig. 8.

V. Conclusions

Maintaining a certain configuration to automatically navigate in complex environments is an important part of the practical application of R-UAV. Due to the assembly and disassembly nature of R-UAV, it is inappropriate to simply regard each UAV module as an agent in MADDPG. In this paper, we propose DMADDG for automatic navigation of R-UAV in complex scenarios. This algorithm introduces a leader decision assignment mechanism to solve the problem of agent disappearance in vehicle assembly. In addition, a collaborative experience reward is developed to constrain the behavior of R-UAV as a combination or individual modules to ensure the reliability of automatic navigation. Both simulation and experimental results demonstrate the effectiveness of the proposed method.

REFERENCES

[1] Wissam Fawaz, Chadi Abou-Rjeily, and Chadi M. Assi. Uav-aided cooperation for fso communication systems. *IEEE Communications Magazine*, 56:70–75, 2018.

[2] Hailong Huang and Andrey V. Savkin. An algorithm of reactive collision free 3-d deployment of networked unmanned aerial vehicles for surveillance and monitoring. *IEEE Transactions on Industrial Informatics*, 16:132–140, 2020.

[3] Bo Zhang, Chi Harold Liu, Jian Tang, Zhiyuan Xu, Jian Ma, and Wendong Wang. Learning-based energy-efficient data collection by unmanned vehicles in smart cities. *IEEE Transactions on Industrial Informatics*, 14:1666–1676, 2018.

[4] Raymond Oung, Frdric Bourgault, Matthew Donovan, and Raffaello D'Andrea. The distributed flight array. *2010 IEEE International Conference on Robotics and Automation*, pages 601–607, 2010.

[5] Michael Duffy, Tony Samaritano, and Advanced Vertical Lift. The lift! project - modular, electric vertical lift system with ground power tether. In *Ahs Forum*, 2015.

[6] David Saldaña, Bruno Gabrich, Guanrui Li, Mark H. Yim, and Vijay R. Kumar. Modquad: The flying modular structure that self-assembles in midair. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 691–698, 2018.

[7] Guanrui Li, Bruno Gabrich, David Saldaña, Jnaneshwar Das, Vijay R. Kumar, and Mark H. Yim. Modquad-vi: A vision-based self-assembling modular quadrotor. *2019 International Conference on Robotics and Automation (ICRA)*, pages 346–352, 2019.

[8] David Saldaña, Parakh M. Gupta, and Vijay R. Kumar. Design and control of aerial modules for inflight self-disassembly. *IEEE Robotics and Automation Letters*, 4:3410–3417, 2019.

[9] Jiawei Xu, Diego S. D'Antonio, and David Saldana. H-modquad: Modular multi-rotors with 4, 5, and 6 controllable dof. In *International Conference on Robotics and Automation*, 2021.

[10] Won Joon Yun A, Soyi Jung B, Joongheon Kim A, and Jae Hyun Kim B. Distributed deep reinforcement learning for autonomous aerial evtol mobility in drone taxi applications. *ICT Express*, 2021.

[11] Han Qie, Dianxi Shi, Tianlong Shen, Xinhai Xu, Yuan Li, and Liujing Wang. Joint optimization of multi-uav target assignment and path planning based on multi-agent reinforcement learning. *IEEE Access*, 7:146264–146272, 2019.

[12] Qingqing Wang, Hong Liu, Kaizhou Gao, and Le Zhang. Improved multi-agent reinforcement learning for path planning-based crowd simulation. *IEEE Access*, 7:73841–73855, 2019.

[13] Dawei Wang, Tingxiang Fan, Tao Han, and Jia Pan. A two-stage reinforcement learning approach for multi-uav collision avoidance under imperfect sensing. *IEEE Robotics and Automation Letters*, 5:3098–3105, 2020.

[14] Yu-Jia Chen, Deng-Kai Chang, and Cheng Zhang. Autonomous tracking using a swarm of uavs: A constrained multi-agent reinforcement learning approach. *IEEE Transactions on Vehicular Technology*, 69:13702–13717, 2020.

[15] Won Joon Yun, Soohyun Park, Joongheon Kim, Myungjae Shin, Soyi Jung, David A. Mohaisen, and Jae-Hyun Kim. Cooperative multiagent deep reinforcement learning for reliable surveillance via autonomous multi-uav control. *IEEE Transactions on Industrial Informatics*, 18:7086–7096, 2022.

[16] Afshin Oroojlooyjadid and Davood Hajinezhad. A review of cooperative multi-agent deep reinforcement learning. *Applied Intelligence*, 53:13677 – 13722, 2019.

[17] Peter Sunehag, Guy Lever, Audrxb, Nas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, and Joel Z Leibo. Value-decomposition networks for cooperative multi-agent learning based on team reward. In *International Conference on Autonomous Agents and Multiagent Systems*, 2018.

[18] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In I Guyon, UV Luxburg, S Bengio, H Wallach, R Fergus, S Vishwanathan, and R Garnett, editors, *ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS 30 (NIPS 2017)*, volume 30 of *Advances in Neural Information Processing Systems*, 2017.

[19] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder de Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Monotonic value function factorisation for deep multi-agent reinforcement learning. *JOURNAL OF MACHINE LEARNING RESEARCH*, 21, 2020.

[20] Alex Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25(2), 2012.