Aerospace Science and Technology 150 (2024) 109166

Contents lists available at ScienceDirect

Aerospace Science and Technology

journal homepage: www.elsevier.com/locate/aescte

Graph-based multi-agent reinforcement learning for large-scale UAVs swarm system control

Bocheng Zhao, Mingying Huo *, Zheng Li, Ze Yu, Naiming Qi

Harbin Institute of Technology, Harbin, 150001, China

ARTICLE INFO

Communicated by Chaoyong Li

Keywords:
Graph neural network
Multi-agent reinforcement learning
Unmanned aerial vehicles swarm
Potential field function
Collision avoidance

ABSTRACT

In this study, a novel graph-embedding technique based on a graph neural network (GNN) is proposed to identify the topology in the motion of a unmanned aerial vehicles (UAV) swarm and quickly obtain local information around each agent. We also propose a model reference reinforcement learning method to learn the potential field function and determine an appropriate strategy for each agent that can satisfy the requirements of collaborative motion and obstacle avoidance for large-scale UAV swarms. First, a new swarm structure is proposed to provide reserved maneuvering space for UAVs during flight. After encoding the obstacle avoidance behavior of multiple UAVs into spatial graphs, a graph attention mechanism (GAT) was employed to extract the dynamic information from them. Consequently, each individual autonomously generate actions based on its local data. Second, a new distributed control algorithm based on multi-agent reinforcement learning (MARL) is proposed to learn the potential field function from the local information. Each individual can repel and cooperate with the target within a short range and attract objects over a long distance. Finally, simulation results demonstrate the effectiveness and superiority of the proposed method, which has great potential for application in online autonomous collaboration.

1. Introduction

In recent years, multiple unmanned aerial vehicles (UAVs) have been widely used in the civilian and military fields [1,2]. Compared to a single aircraft, multiple UAVs with stronger capabilities and larger coverage are more flexible and robust [3–5]. The key issue in the research on multi-UAV systems is the obstacle avoidance problem of UAVs. Each member of the UAV swarm needs to avoid obstacles (such as terrain obstacles, birds, and enemy swarms) as well as other nearby UAVs in the swarm [6]. In addition, if each UAV chooses actions independently, more collisions may occur in that actions of the members within the swarm affect others. The working environment of UAVs is typically complex and collision problems inevitably occur, which may also cause damage to them [7]. Extensive research has been conducted on motion control methods that incorporate different constraints, including intelligent algorithms [8], stochastic planning [9], online optimal control technology [10,11], model predictive control method [12], reinforcement learning (RL) [13,14], and artificial potential field method (APF) [15]. Li et al. introduced a grouped particle swarm optimization algorithm which enables the optimization of multi-segment paths [8]. Zhang et al. introduced a motion planning strategy that conceptualized motion planning as open-loop optimization problem which achieve precise trajectory tracking for space robots while considering various physical constraints [16]. Ma et al. proposed a nonlinear model predictive control law leveraging deep learning to achieve cost-effective system feedback control [17]. Wei et al. validated their proposed multi-level control strategy through empirical experiments, demonstrating its successful implementation in achieving autonomous, collision-free assembly of multiple spacecraft simulators [18]. Jiang et al. proposed an adaptive controller based on RL, which dynamically generates control actions and integrates trajectory planning into the overall dynamic solution [19]. Wei et al. investigated the planar rendezvous and docking process based on APF between the chaser spacecraft simulator and the rotating target spacecraft simulator [20]. Unfortunately, a slice of existing approaches suffers from significant computational complexity, prolonged response times, and a reliance on global environmental data for generating collision-free trajectories when employed in the context of cooperative motion control and obstacle avoidance for large-scale UAVs

* Corresponding author.
E-mail addresses: 22b918135@stu.hit.edu.cn (B. Zhao), huomingying@hit.edu.cn (M. Huo), li779310083@gmail.com (Z. Li), yuze@hit.edu.cn (Z. Yu), qinmok@163.com (N. Qi).

https://doi.org/10.1016/j.ast.2024.109166
Received 24 February 2024; Received in revised form 6 April 2024; Accepted 23 April 2024
Available online 26 April 2024

[21,22]. In addition, with the need for practical applications, the number of individuals in UAV swarms has increased significantly, resulting in large-scale UAVs swarm operations which requires the use of distributed execution and scalable algorithms based on an understanding of global information, issue in the above centralized control methods no longer feasible. Global planning grows exponentially with the number of UAVs and the environmental size [23] which is particularly evident in large-scale UAV swarm systems of hundreds of thousands [24,25]. Commonly used methods have good control effects in a certain number of multi-UAV formations and obstacle avoidance controls, but are no longer suitable for online obstacle avoidance and collaborative movement of large-scale UAV swarm systems [22].

Furthermore, when completing complex tasks, it is necessary to dynamically switch between formations. Therefore, the collaborative control of multiple UAVs not only relies on obstacle avoidance methods but also requires formation control as the basis [26,27]. The collaborative control of multiple UAVs relies on obstacle avoidance methods and requires formation control as the basis. Research in the field of large-scale UAV swarm control can be divided into two main parts based on the degree of autonomy of UAVs. Partly because many companies use UAVs in light shows, which are automated systems designed for live performances and can control thousands of UAVs [28,29]. However, these UAV swarm control methods ensure safety of UAV by tracking predetermined trajectories. If any one of the UAVs malfunctions, the entire fleet is affected. In addition, the control systems usually adopt a central control architecture that requires each individual to communicate with the ground station, resulting in a large number of calculations and making it impossible to perform online autonomous swarm control [30].

The main challenges faced by large-scale UAV swarm collaborative trajectory planning and obstacle avoidance are as follows. 1) Online control of UAV swarms to ensure real-time obstacle avoidance with a high success rate. 2) Stable formation maintenance and rapid formation reconstruction after obstacle avoidance movements in large-scale scalable swarms. 3) Each UAV can act independently based on the local information obtained, while ensuring the adaptability of the swarm to dynamic random environments and the consistency of the swarm structure.

Multiagent reinforcement learning (MARL) is a promising multi-UAV collision avoidance method that models the problem as a decentralized partially observable Markov decision process (Dec-POMDP) by employing a centralized training and distribution method of execution (CTDE) [21]. Specifically, MARL trains multiple agents in a centralized manner by leveraging local and global information that is not available when agents make decisions during execution [31]. All the agents independently performed decision-making algorithms based on the same critical network, which can transfer the complexity of a problem to offline training and render online execution lightweight [32].

Graph neural network (GNN) is effective in representing the known structures of arbitrary relational systems. A GNN can combine the powerful learning ability of neural networks with graph interpretability. An increasing number of studies have used it to solve heuristic UAV swarm problems, such as motion planning [33,34], coverage [35], and perimeter defense [36]. GNNs can be used to extract features and reduce dimensions by rapidly processing global information in the situations of large-scale UAVs swarm and representing all elements in a single spatial graph with only local connections. In this study, we propose a novel graph-based multi-agent reinforcement learning algorithm based on potential functions to solve the challenges faced by large-scale UAV swarm collaborative trajectory planning and obstacle avoidance. The contributions of this study can be summarized as follows:

1) Different from [35] which encodes multi-robots as a discrete graph, we encode the multi-UAV problem data into a continuous spatial graph where UAVs, obstacles, and reference points are nodes, and the relationships between UAVs, UAVs and obstacles, and UAVs and reference points are edges. A position-based graph attention (PGAT) mechanism that adaptively learns the attention weight coefficient of the graph is proposed, which avoids excess learning parameters by reducing the influence of individuals that are distant and realizes efficient representation in a dynamic environment through the dynamic update of nodes in the graph.
2) To reserve a certain maneuvering space for UAV motion, a new group structure suitable for large-scale UAV motion control was introduced. Compared to the rigid group structure in [22], the structure of UAVs proposed in this study exhibits elevated flexibility by allowing dynamic adjustments in both the number of UAVs with in each cluster and the space between adjacent clusters. In addition, we propose a distributed control architecture to group large-scale swarms for parallel processing. Different from [37], each UAV operates independently in this paper, relying solely on locally acquired information which reduce the dependence on global information sources and the requirements for communication capabilities.
3) A MARL-based UAVs swarm control algorithm is proposed, which we call PGAT-**M**ulti-**A**gent **P**otential **F**ield function **L**earning algorithm (PGAT-MAPFL). Instead of approach which relies on curriculum learning for training and gradually scales up to larger configurations in [38], our novel contribution lies in the utilization of GAT which is employed to learn the potential field functions, thereby enhancing the adaptability of UAV swarms to dynamic random environments and the consistency of actions. To simplify the MARL model and improve the training efficiency and effect based on the equivalence between agents, we designed a collaborative learning model framework such that different agents can share actor networks, reviewer networks, and interaction experiences. The feasibility of the proposed PGAT-MAPL algorithm is verified through simulations.

The remainder of this study is organized as follows. Section 2 introduces the problem statement and modeling in this study. Section 3 presents the proposed method. Section 4 demonstrates the experiments of training process and the performance in tests of the proposed method. Ultimately, Section 5 draws the conclusion of this study.

2. Preliminaries and modeling

2.1. Problem statement

This study aims to conduct an in-depth investigation of the **m**otion **c**ontrol and **o**bstacle **a**voidance of **l**arge-scale UAV swarms without global status (MCOAL). Distributed control can control swarms when local information is obtained, and its advantages are discussed in the Introduction. Assuming each UAV can detect the status of other UAVs within its maximum sensing range via a wireless communication channel or airborne radar, the goal was to maintain a certain distance from other UAVs in the group and avoid contact with other UAVs in the group. UAVs need to avoid being too close or too far, try to maintain the formation at the same speed, and quickly restore the formation after avoiding collisions with obstacles. The control objective of this system can be formulated as:

$$\lim_{t\to+\infty} E = \left\{E_u + E_c + E_m\right\}_{\min} \tag{1}$$

where E represents the total energy of the UAV swarm, E_u represents the energy for avoiding collisions with other UAVs in the swarm, E_c represents the energy for avoiding collisions with obstacles, E_m represents the energy for maintain swarm formation. When$t \to +\infty$ and E reaches its minimum, the entire system achieves its optimal control goal. Simultaneously, the UAV group maintains a consistent speed after avoiding obstacles [22].

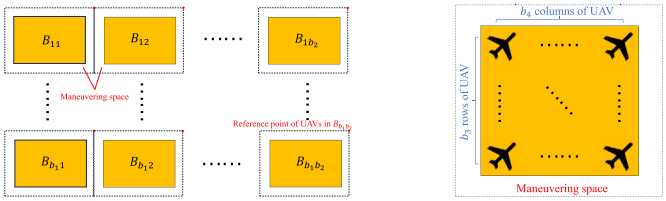

Fig. 1. UAV structure at ideal state, where b_1 and b_2 are the maximum number of rows and columns containing sub-matrices in the swarm, respectively, and b_3 and b_4 are the maximum number of columns and rows containing UAVs in sub-matrices B_{ij}, respectively.

2.2. Structure of UAVs swarm

We propose a formation structure with maneuverable space that utilizes parallelism to group large-scale UAV swarms by dividing the large-scale UAV swarm into several groups and setting the reference point of each group to move toward the desired location and maintain the formation. This structure preserved space for obstacle avoidance and maneuvering. Each UAV only needs to obtain the statuses of the other UAVs within its interaction radius d_τ. In other words, any UAV requires only local information in the current state to output actions. The entire UAVs swarm includes a certain number of sub-matrices $\boldsymbol{B}_{ij}$ that represent a small group of UAVs, as shown in Fig. 1.

$\tau_{mn}^{B_{ij}}$ denotes UAV in the mth row and nth column of $\boldsymbol{B}_{ij}$, and its position coordinate can be expressed as $\boldsymbol{p}_{mn}^{B_{ij}} = [x_{mn}^{B_{ij}}, y_{mn}^{B_{ij}}]^\top$, $\boldsymbol{p}_r^{B_{ij}}$ denotes the position of reference point of $\boldsymbol{B}_{ij}$. We considered a simplified dynamic model for a UAV, which can be expressed as

$$\begin{bmatrix} \dot{x}_{mn}^{B_{ij}} \\ \dot{y}_{mn}^{B_{ij}} \\ \dot{v}_{mn}^{B_{ij}} \\ \dot{\psi}_{mn}^{B_{ij}} \end{bmatrix} = \begin{bmatrix} v_{mn}^{B_{ij}} \cos(\psi_{mn}^{B_{ij}}) \\ v_{mn}^{B_{ij}} \sin(\psi_{mn}^{B_{ij}}) \\ u_{mn}^{vB_{ij}} \\ u_{mn}^{\psi B_{ij}} \end{bmatrix} \tag{2}$$

where $v_{mn}^{B_{ij}}$, $\psi_{mn}^{B_{ij}}$ and $\boldsymbol{u}_{mn}^{B_{ij}} = [u_{mn}^{vB_{ij}}, u_{mn}^{\psi B_{ij}}]^\top$ are the yaw angle, the ground speed, and the control of UAV $\tau_{mn}^{B_{ij}}$, respectively. The first derivative of $p_{mn}^{B_{ij}}$ is expressed as $\boldsymbol{q}_{mn}^{B_{ij}} = \dot{\boldsymbol{p}}_{mn}^{B_{ij}} = [\dot{x}_{mn}^{B_{ij}}, \dot{y}_{mn}^{B_{ij}}]^\top$, and the second derivative of $\boldsymbol{p}_{mn}^{B_{ij}}$ can be expressed as

$$\ddot{\boldsymbol{p}}_{mn}^{B_{ij}} = [\ddot{x}_{mn}^{B_{ij}}, \ddot{y}_{mn}^{B_{ij}}]^\top = \begin{bmatrix} \cos(\psi_{mn}^{B_{ij}}) & -v_{mn}^{B_{ij}} \sin(\psi_{mn}^{B_{ij}}) \\ \sin(\psi_{mn}^{B_{ij}}) & v_{mn}^{B_{ij}} \cos(\psi_{mn}^{B_{ij}}) \end{bmatrix} \begin{bmatrix} u_{mn}^{vB_{ij}} \\ u_{mn}^{\psi B_{ij}} \end{bmatrix} \tag{3}$$

Let $\begin{bmatrix} \cos(\psi_{mn}^{B_{ij}}) & -v_{mn}^{B_{ij}} \sin(\psi_{mn}^{B_{ij}}) \\ \sin(\psi_{mn}^{B_{ij}}) & v_{mn}^{B_{ij}} \cos(\psi_{mn}^{B_{ij}}) \end{bmatrix} = \boldsymbol{H}_{mn}^{B_{ij}}$, the new control of system $\boldsymbol{w}_{mn}^{B_{ij}}$ follows

$$\boldsymbol{u}_{mn}^{B_{ij}} = \boldsymbol{H}_{mn}^{B_{ij}{}^{-1}} \boldsymbol{w}_{mn}^{B_{ij}} \tag{4}$$

Therefore, (2) can be simplified as a second-order integral system expressed as

$$\begin{cases} \dot{\boldsymbol{p}}_{mn}^{B_{ij}} = \boldsymbol{q}_{mn}^{B_{ij}} \\ \dot{\boldsymbol{q}}_{mn}^{B_{ij}} = \boldsymbol{w}_{mn}^{B_{ij}} \end{cases} \tag{5}$$

2.3. Graph construction of MCOAL

In our swarm scenario, a dynamic group of UAVs flies at a fixed altitude, and the UAVs (also called agents) in each group follow the reference point of the group. The behavior of a reference point is determined by the external commands generated by a specific task. We focus on designing the collision-free movement of UAV swarms, where the goal of each agent is to maintain a certain distance from other agents in the swarm while avoiding collisions with obstacles. Therefore, we assume that each agent is capable of avoiding other nearby agents (called neighbors) as well as obstacles within its interaction radius d_τ, as shown in Fig. 2, where $\boldsymbol{p}_{k,k_1} = \boldsymbol{p}_k - \boldsymbol{p}_{k_1}$ is the relative position of agent k and agent k_1, $\boldsymbol{p}_{k,r} = \boldsymbol{p}_k - \boldsymbol{p}_r$ is the relative position of agent k and reference point r, d_o is the repulsive radius of the obstacle o, $\boldsymbol{p}_{k,o} = \boldsymbol{p}_k - \boldsymbol{p}_o$ is the relative position of the agent k and the obstacle o.

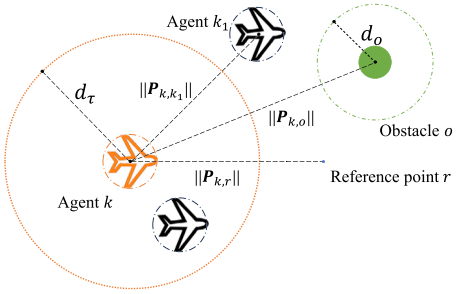

Fig. 2. Schematic diagram of local status of each UAV.

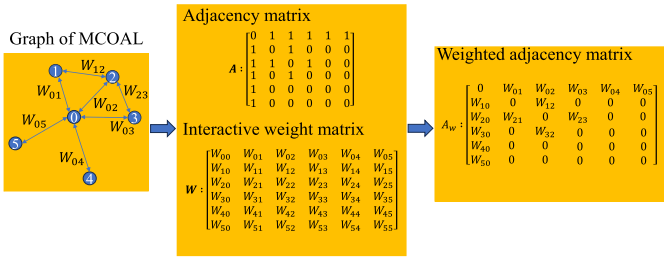

Fig. 3. Generation process of weighted adjacency matrix in MCOAL. $\boldsymbol{A}_W = A \odot W$, W_{ij} is the weight represented by the edge between UAVi and UAVj.

We developed a parameterized current system state in which the parts of the UAVs within the group, reference points, and obstacles were part of the computational diagram. Each UAV, reference point, and obstacle in the swarm is considered to be a node in the graph, and the graph topology changes during execution owing to the movement of the UAV. Unlike coverage problems, such as [39], we do not need to provide the relative distances from all reference points to all agents. There are three types of edges in the graph: 1) mapping the edges between UAVs within a certain range to indicate the positional relationship between UAVs, 2) mapping the edges between UAVs and obstacles within a certain range to indicate the positional relationship between the UAV and the obstacle, and 3) mapping the action edge between the UAV and the reference point, which indicates the UAV's ability to move to the desired location. All three types of edges were undirected. Assume that UAVs can communicate with their neighbors. Each agent can obtain the information of any node in a graph composed of MCOAL through communication, and its connectivity can be represented by the adjacency matrix A. When an edge exists between two agents, the value of the corresponding element in the adjacency matrix is 1; otherwise, it is 0. Owing to the constraints of the swarm collaborative control task, there are interactive relationships such as repulsion and attraction between agents. The specific calculation method is introduced in Section 3.1. Based on the relationship between these agents, the weighted adjacency matrix $\boldsymbol{A}_W$ is obtained, and the construction process is shown in Fig. 3.

We denote the set of vertex features as $\boldsymbol{V}$, the set of edge features as $\boldsymbol{E}$, and the state of the system as $\boldsymbol{G} = \{\boldsymbol{E}, \boldsymbol{V}\}$. The set of neighbors of agent k can be expressed as $\boldsymbol{N}_k = \{l \in \boldsymbol{V}_a : \|\boldsymbol{p}_k - \boldsymbol{p}_l\| \le d_\tau, l \ne k\}$, where $\boldsymbol{V}_a$ is the node set of agents. The neighbor obstacle set of agent k can be expressed as $\boldsymbol{O}_k = \{o \in \boldsymbol{V}_o : \|\boldsymbol{p}_k - \boldsymbol{p}_o\| \le d_\tau\}$, where $\boldsymbol{V}_o$ is a node set of obstacles.

2.4. Dec-POMDP of UAV swarm control

We model MCOAL as a decentralized partially observable Markov decision process (Dec-POMDP) [40].

2.4.1. State and action

Assuming that a UAV swarm flies at a fixed altitude, the status of each UAV can be represented by the tuple $\xi_k := (\xi_\tau, \xi_r, \boldsymbol{\xi}_{\boldsymbol{N}_k}, \boldsymbol{\xi}_{\boldsymbol{O}_k})$. Among them, $\xi_\tau = (\boldsymbol{p}_\tau, \boldsymbol{q}_\tau)$ represents the state of the UAV itself, ξ_r represents the state of the reference point, $\boldsymbol{\xi}_{\boldsymbol{N}_k} := \{\xi_l | l \in \boldsymbol{N}_k\}$ represents the local state of neighbors, and $\boldsymbol{\xi}_{\boldsymbol{O}_k} := \{\xi_o | o \in \boldsymbol{O}_k\}$ represents the local state of obstacle. The global system state is expressed as $s := (\boldsymbol{\xi})$. The state-transition function of the system is defined in (2). The control of agent k is performed by using the control quantity $\boldsymbol{w}_k = \ddot{\boldsymbol{p}}_k$ in continuous spaces. The actions of the entire UAVs swarm are represented as a set of amounts controlled by each UAV $a := (\boldsymbol{w})$.

2.4.2. Reward function

The goal of MCOAL is to learn a collision-free swarming strategy for the distributed control of each UAV in a large-scale UAV swarm. From the perspective of each follower, swarm control has three goals: reducing the distance to the desired location (where the desired location is determined by the reference point), maintaining a certain distance from its neighbors, and avoiding collisions with obstacles. The reward for proximity is set to motivate agent k to approach the desired location.

$$R_k^c = -C_1 \left\| \boldsymbol{p}_{k,r} \right\| \tag{6}$$

where C_1 is the tunning parameter. We define the formation-keeping reward r_f to incentivize UAVs to keep an appropriate distance from their neighbors and avoid getting too far apart or colliding.

$$R_k^f = \begin{cases} -C_2, & \left\| \boldsymbol{p}_{k,l} \right\| \le d_c \\ -C_3 \left\| \boldsymbol{p}_{k,l} \right\|, & \left\| \boldsymbol{p}_{k,l} \right\| > d_{max} \end{cases} \tag{7}$$

where C_2 and C_3 are the tunning parameters, $\left\| \boldsymbol{p}_{k,l} \right\|$ is the distance between UAV k and its neighbor l. d_c and d_{max} are the minimum safe distances between swarms and the maximum distance required to maintain the formation, respectively. Defining the Collision Penalty with Obstacles

$$R_k^o = C_4(-d_o + \left\| \boldsymbol{p}_{k,o} \right\|), \left\| \boldsymbol{p}_{k,o} \right\| \le d_o \tag{8}$$

where C_4 is the tunning parameter and d_o is the minimum safe distance between agent k and obstacle o. Ultimately, the reward function for agent k can be defined as

$$R_k = R_k^c + R_k^o + \sum_{l \in \boldsymbol{N}_k} R_k^f \tag{9}$$

The rewards earned by the entire swarm in an environment can be expressed as the sum of the rewards earned by each UAV.

$$R = \sum_{i=1,j=1}^{b_1,b_2} \sum_{m=1,n=1}^{b_3,b_4} R_{\tau_{mn}^{B_{ij}}} \tag{10}$$

where b_3 and b_4 denote the maximum number of columns and maximum number of sub-matrices in UAVs swarm, respectively.

3. Proposed algorithm

The MARL method [41,38], artificial potential field (APF) method [42], and improved methods [22] are commonly used for UAV swarm motion control and obstacle avoidance. To overcome the inherent shortcomings of the APF being unreachable, falling into local optima, and improving the obstacle avoidance effect, we propose a new PGAT-MAPFL method which includes the advantages of the methods. Sections 3.1 and 3.3 describe the details of the two core components of PGAT-MAPFL, namely, the control strategy based on the potential field function and the policy learning of the PGAT-MAPFL algorithm, by introducing the attention mechanism.

3.1. Potential field function design

In a multiagent system, as the number of agents increases, the system's state and action space dimensions grow exponentially, which provides high performance and efficient collision-free control for large-scale UAV swarms [43]. Strategy learning presents significant challenges. To solve this problem, we propose a method for learning the coefficients of a potential function to improve algorithm performance. The potential field functions were designed based on the interaction between the UAVs, obstacle avoidance, and maintenance of the UAV group structure during movement. Since the Euclidean norm is not differentiable at zero, we use σ-norm to construct a nonnegative smooth potential function between two nodes [37]. σ-norm can be expressed as

$$\|z\|_\sigma = \frac{1}{\zeta}[\sqrt{1 + \zeta \|z\|^2} - 1] \tag{11}$$

where $\zeta > 0$ denotes a parameter. The potential field construction in UAV swarm consists of three parts: the potential field of swarm structure maintenance, potential field of motion control, and potential field of collision avoidance.

3.1.1. Potential field of swarm structure maintenance

The potential field function used to maintain the desired distance between UAV k and UAV l is defined as follows:

$$\phi_u(\left\| \boldsymbol{p}_{k,l} \right\|_\sigma) = K_1 \int_{\left\| d_i \right\|_\sigma}^{\left\| p_{k,l} \right\|_\sigma} \rho_\tau(\frac{x}{\left\| d_\tau \right\|_\sigma}) \frac{x - \left\| d_i \right\|_\sigma}{\sqrt{1 - (x - \left\| d_i \right\|)^2}} dx \tag{12}$$

where K_1 is a parameter used to adjust the strength of the potential field. When two UAVs maintain an ideal distance $\left\| d_i \right\|$, their potential field function tends towards the global minimum, and the formation-keeping function ρ_τ can be expressed as

$$\rho_\tau(y) = \begin{cases} 1, & y \in [0, y_d) \\ \frac{1}{2}[1 + \cos(\pi \frac{y - y_d}{1 - y_d})], & y \in [y_d, 1] \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

where $y_d \in (0, 1)$. According to (12), the potential field between UAV k and UAV l has different manifestations under the following three conditions:

1) When $\left\| \boldsymbol{p}_{k,l} \right\|_\sigma \ge \left\| d_\tau \right\|_\sigma$, the potential field between UAV k and UAV l is 0.
2) When $\left\| \boldsymbol{p}_{k,l} \right\|_\sigma < \left\| d_{\text{ideal}} \right\|_\sigma$, UAV l excludes UAV k.
3) While $\left\| d_{\text{ideal}} \right\|_\sigma < \left\| \boldsymbol{p}_{k,l} \right\|_\sigma < \left\| d_\tau \right\|_\sigma$, UAV l attracts UAV k.

3.1.2. Potential field of motion control

The movement of each group of UAVs is defined by the attractive potential field formed between the reference point and group of UAVs. The potential field function of the motion control is defined as

$$\phi_r(\left\| \boldsymbol{p}_{k,r} \right\|_\sigma) = K_2 \int_0^{\left\| p_{k,r} \right\|_\sigma} x dx \tag{14}$$

where K_2 is a parameter used to adjust the strength of the potential field. When $\left\| \boldsymbol{p}_{k,r} \right\|_\sigma \to 0$, the potential field between the reference point and the group of UAVs tends to 0.

3.1.3. Potential field of collision avoidance

The potential field function between UAV k and obstacle can be defined as

$$\phi_o(\|\boldsymbol{p}_{k,o}\|_\sigma) = K_3 \int_{\|d_o\|_\sigma}^{\|\boldsymbol{p}_{k,o}\|_\sigma} \rho_o\left(\frac{x}{\|d_o\|_\sigma}\right)\frac{x-\|d_o\|_\sigma}{x^3}dx \tag{15}$$

where the obstacle collision avoidance function ρ_o can be expressed as follows:

$$\rho_o(y) = \begin{cases} 1, & y \in [0,1] \\ 0, & \text{otherwise} \end{cases} \tag{16}$$

According to (15), the potential field between UAV k and obstacle o exhibits different manifestations under the following two conditions:

1) When $\|\boldsymbol{p}_{k,o}\|_\sigma \to 0$, the potential field between the obstacle and the UAV tends to ∞.
2) When $\|\boldsymbol{p}_{k,o}\|_\sigma > \|d_o\|_\sigma$, the potential field between the obstacle and the UAV tends to 0.

3.2. Control algorithm of swarm

To achieve the goals of motion control and obstacle avoidance of UAV swarm, we designed a group control algorithm for each UAV that considers UAV group structure maintenance and obstacle avoidance. The control of agent k can be expressed as follows:

$$\begin{aligned}\boldsymbol{w}_k = &-\sum_{l\in\boldsymbol{N}_k}\frac{\partial\phi_u(\|\boldsymbol{p}_{k,l}\|_\sigma)}{\partial(\boldsymbol{p}_{k,l})} - \sum_{o\in\boldsymbol{O}_k}\frac{\partial\phi_o(\|\boldsymbol{p}_{k,o}\|_\sigma)}{\partial(\boldsymbol{p}_{k,o})} \\ &-\frac{\partial\phi_r(\|\boldsymbol{p}_{k,r}\|_\sigma)}{\partial(\boldsymbol{p}_{k,r})} - K_5\boldsymbol{q}_{k,r} + K_6\sum_{l\in\boldsymbol{N}_k}\boldsymbol{q}_{k,l}\end{aligned} \tag{17}$$

where K_5 and K_6 are coefficients.

3.3. Position-based graph attention network

GNN and its derived versions can learn the embedded representations of graphs [44] and have been successfully applied to link prediction [45] and node classification tasks. However, these methods mainly focus on the embedded representation of a single node and essentially adopt a flat processing method. Therefore, they are not efficient for processing data with a hierarchical structure. In this study, we aim to learn global topological information from local information. Therefore, we used a multistage GNN. First, an obstacle model is established using a position-based graph attention network (PGAT), where the positions of neighbor agents and obstacles are used as the input features. Second, given the local obstacle information, distributed UAV motion control is performed through multiagent reinforcement learning. In the obstacle modeling stage, other UAVs and the given local obstacles within a certain range around each agent are regarded as a set, and a graph of changes over time is provided. Unlike traditional GNN methods that consider only static graphs, PGAT performs well in dynamic environments where mobile nodes can join or leave the system at any time. PGAT extracts hidden node embedding features by iteratively exchanging information with neighboring nodes. The node embedding characteristics of each node are a function of the node's local information and actions (control quantities), and the contributions of other nodes are used to characterize the underlying network structure. Specifically, in the context of GAT, three fundamental elements are involved: Query, Key, and Value [46]. We consider the location information of the current agent ($\boldsymbol{p}_k$) as the Query, the location information of neighboring agents ($\boldsymbol{p}_l$) and the location information of neighboring obstacles ($\boldsymbol{p}_o$) as the Key and the high-dimensional feature of observation information (f_{kl} and f_{ko}) as the Value. The attention weights of neighbor obstacles and neighbor agents of agent k can be expressed as

$$\begin{aligned}W_{kl} &= \begin{cases}\exp(-c_a^W\|\boldsymbol{p}_k-\boldsymbol{p}_l\|), & l\in\boldsymbol{N}_k \\ 0, & l\notin\boldsymbol{N}_k\end{cases} \\ W_{ko} &= \begin{cases}\exp(-c_o^W\|\boldsymbol{p}_k-\boldsymbol{p}_o\|), & l\in\boldsymbol{O}_k \\ 0, & o\notin\boldsymbol{O}_k\end{cases}\end{aligned} \tag{18}$$

where c_a^W and c_o^W are the attenuation coefficients. The attention weight is normalized by the softmax function expressed as

$$\begin{aligned}\gamma_{kl} &= \text{softmax}(W_{kl}) = \frac{\exp(W_{kl})}{\sum_{l\in\boldsymbol{N}_k}\exp(W_{kl})} \\ \gamma_{ko} &= \text{softmax}(W_{ko}) = \frac{\exp(W_{ko})}{\sum_{o\in\boldsymbol{O}_k}\exp(W_{ko})}\end{aligned} \tag{19}$$

The feature aggregation of neighbor obstacles and neighbor agents of agent k can be expressed as follows:

$$\begin{aligned}f_k^a &= \sum_{l\in\boldsymbol{N}_k}\gamma_{kl}f_{kl} \\ f_k^o &= \sum_{o\in\boldsymbol{O}_k}\gamma_{ko}f_{ko}\end{aligned} \tag{20}$$

where f_{kl} and f_{ko} represent the high-dimensional features of the neighbor agent and the neighbor obstacles, respectively. The output features based on PGAT can be obtained by dimensional transformation and concatenation.

$$f_k' = \delta(f_k^a\boldsymbol{F_a}||f_k^o\boldsymbol{F_o}||f_k^r\boldsymbol{F_r}) \tag{21}$$

where $\boldsymbol{F_a}$, $\boldsymbol{F_o}$, $\boldsymbol{F_r}$ are the weights of the networks; $\delta(\cdot)$ is the ReLu function; $(\cdot||\cdot)$ is the concatenation operation; f_k^r is the aggregated feature of the reference point of agent k. The calculation framework of the PGAT is illustrated in Fig. 4.

Since the farther away neighbor agents or neighbor obstacles are, the smaller the attention weight, PGAT can reduce the influence of remote nodes on the decision-making of each agent. Considering that the output characteristics and network weights of PGAT are independent of the scales of the agent and obstacle, PGAT was adapted for MCOAL.

3.4. Structure of PGAT-MAPFL

In the APF method, the influence of local minimum is extremely serious. Similar to the method in [47], [48] and [49] where virtual target points are utilized to modify the force balance at a specific position, the introduced reference point of the cluster could mitigate the local minimum in this paper. In addition, PGAT-MAPFL adopts Actor-Critic architecture to train and output the dynamically adjusted coefficient of the potential function through the neural network. Rather than being manually fixed at a constant value, it becomes a function dependent on the relative position of the objects, which reduces the influence of local minimum while avoiding the suboptimality of manual parameter tuning with the lack of expert experience. The critic network was built using a traditional convolution and multilayer perceptron. The first layer is a one-dimensional convolution layer, and the last two layers are fully connected, as shown in the **blue part** of Fig. 5. PGAT was introduced in the actor network to change the attention-level information of neighboring nodes by changing their weights. After the *tanh* function limits the output to the interval of $(-1,1)$, we introduce the potential field function built in Section 3.1 into the last layer of the actor network to learn the parameter K Then, the control variable $\boldsymbol{w}$ of each UAV can be output by (17). The structure of the actor network is indicated in blue in Fig. 5. During the training process, the global system state s was known [50].

In the reinforcement learning module, unlike some multi-agent reinforcement learning algorithms, PGAT-MAPFL learns only a shared central critic network, whereas each agent within the group learns its own actor network (Fig. 6). Store the data in replay buffer $\boldsymbol{D}$ with tuple $(s, s', \boldsymbol{A}_W, \boldsymbol{A}'_W, \xi, \xi', R, a)$, where s', $\boldsymbol{A}'_W$, ξ' are the local states of

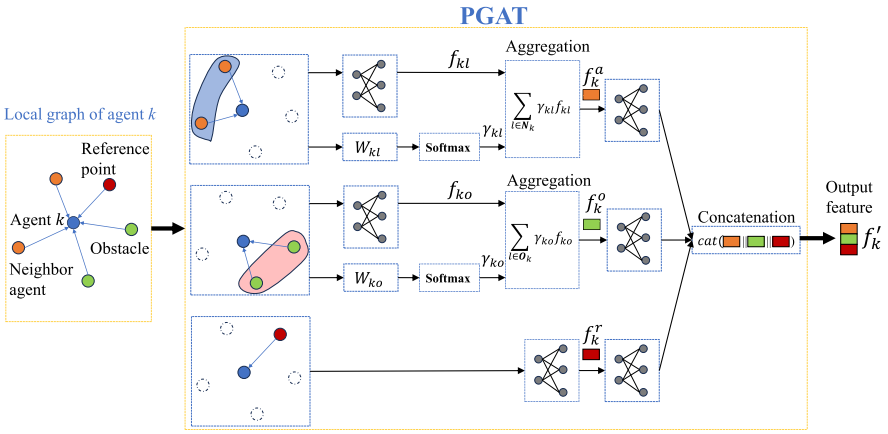

Fig. 4. Calculation framework of the PGAT.

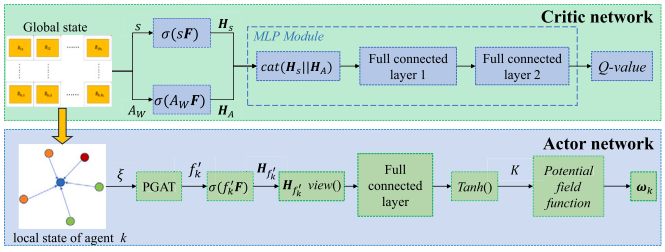

Fig. 5. The Actor network and Critic network, where $\boldsymbol{F}$ is the weight of neural network layer, $\boldsymbol{H}_{\cdot} = \delta(\cdot \boldsymbol{F})$ is the high-dimensional representation of the properties, $view(\cdot)$ is the straighten function which can transfer the tensor into a vector. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

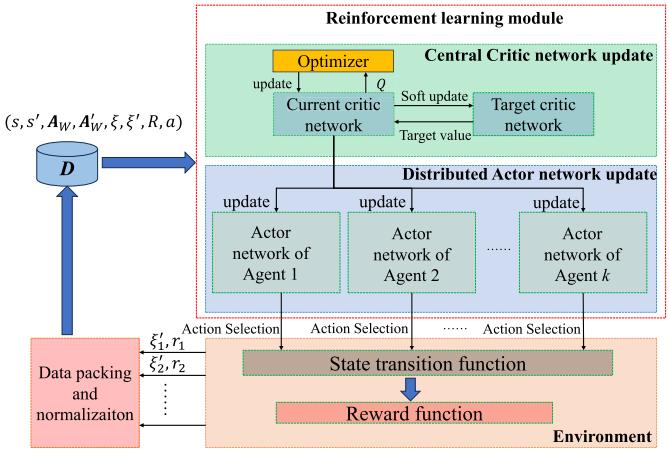

Fig. 6. Structure of PGAT-MAPFL algorithm.

all agents, the weighted adjacency matrixs of all agents and the global state after taking action a, respectively.

PGAT-MAPFL adopts centralized and distributed execution, as shown in Fig. 6. Owing to the distinct observation perspectives of the central critic and the distributed executor, a neural network structure with shared parameters cannot be employed. In addition, the losses of critics and participants were calculated separately. PGAT-MAPFL uses centralized dominance estimation technology for central training, observes the observations and movements of all UAVs, and fully considers the status of all UAVs in the environment. Traditional MARL algorithms (such as MADDPG [51] and MATD3 [52], etc.) can only minimize the local loss of the current agent during the strategy optimization process and cannot guarantee the minimum global loss of the entire system. The global reward method was used to optimize the swarm collaborative control strategy. The optimization goal can be written as

$$\boldsymbol{J}(\theta^{\lambda}) = \mathbb{E}_{s\sim s^{\lambda}, a\sim\lambda}(\sum_{t=0}^{T} \eta \boldsymbol{R}^{t}(s^{t}, a^{t})) \tag{22}$$

where s^{λ} is the state distribution of actor λ, θ^{λ} is a parameter of the actor network, and $\eta \in [0, 1)$ is a constant discount factor.

In the central critic network, Q can be expressed as

$$Q^{\lambda}(s, a) = \mathbb{E}_{s'}[R(s, a) + \eta \mathbb{E}_{a'\sim\lambda} Q^{\lambda}(s', a')] \tag{23}$$

By randomly sampling a small batch from $\boldsymbol{D}$ and updating the actor network using Q values from the critic network [52], the gradient can be expressed as

$$\nabla_{\theta^{\lambda}} \boldsymbol{J}(\theta^{\lambda}) = \mathbb{E}_{(s,\xi,a)\sim\boldsymbol{D}}[\nabla_{\theta^{\lambda}} \lambda(\xi_k|\theta^{\lambda}) \nabla_{\boldsymbol{w}_k} Q^{\lambda}(s, a|\theta^{Q})|_{\boldsymbol{w}_k=\lambda(\xi_k|\theta^{\lambda})}] \tag{24}$$

According to (23), the target Q value is defined as follows:

$$Q_{\text{tar}} = R(s, a) + \eta Q^{\widetilde{\lambda}}(s', a')|_{a'=\widetilde{\lambda}(\xi_k')} \tag{25}$$

where $\widetilde{\lambda}$ is the target actor network, $Q^{\widetilde{\lambda}}$ is the target critic network, the parameters of which can be updated by minimizing the following loss functions:

$$\boldsymbol{L}(\theta^{Q^{\lambda}}) = \mathbb{E}_{(s,a,R,s')\sim\boldsymbol{D}}[Q^{\lambda}(s, a) - (r + \eta Q^{\widetilde{\lambda}}(s', a')|_{a'=\widetilde{\lambda}(\xi_k')})] \tag{26}$$

According to (24) and (26), we complete the training of the actor and critic networks, and obtain a strategy to minimize the loss of the system. The parameters of the target actor and critic networks can be updated using the following formulae:

$$\begin{cases} \theta^{\widetilde{\lambda}} \leftarrow \mu^{\lambda}\theta^{\lambda} + (1 - \mu^{\lambda})\theta^{\widetilde{\lambda}} \\ \theta^{\widetilde{Q}} \leftarrow \mu^{Q}\theta^{Q} + (1 - \mu^{Q})\theta^{\widetilde{Q}} \end{cases} \tag{27}$$

where $\theta^{\widetilde{\lambda}}$ and $\theta^{\widetilde{Q}}$ are the parameters of the target actor and critic network, respectively. $\mu^{\lambda} < 1$ and $\mu^{Q} < 1$ are update rates.

In MCOAL, the system loss can characterize the performance of agent cooperation during the process of completing stable flocking control and obstacle avoidance tasks. Compared with the traditional MARL algorithm, the strategy evaluation mechanism of PGAT-MAPFL algorithm can make use of the system loss to construct a global reward and reflect the control error of each agent. Therefore, the PGAT-MAPFL algorithm can optimize the control strategy of each agent or even the entire swarm control system by evaluating the behavior strategy of each agent using global rewards. The control algorithm is presented in Algorithm 1, where β denotes standard deviation.

Algorithm 1 PGAT-MAPFL algorithm.

Input: Maximum capacity of replay buffer $\boldsymbol{D}_{\max}$; Network update frequency α_f; Maximum training episode in each stage $\alpha_{e_{\max}}$; batch size of training α_b; Maximum time step in each episode α_t

Initialize parameters θ^{λ} and θ^{Q}.
Empty replay buffer $\boldsymbol{D}$
for $z = 1$ to $\alpha_{e_{\max}}$ **do**
  Initialize the system state s randomly.
  for $t = 1$ to α_t **do**
    Select actions using the actor network of each agent and Gaussian exploration noise.
    $w_k = \lambda(\xi_k)|\theta^{\lambda} + \mathbb{N}(0, \beta^2)$
    Update ξ_k by using (2):
    Calculate ξ_k' and s'
    Calculate $\boldsymbol{R}$ using (10)
    Store $(s, s', \xi, \xi', \boldsymbol{R}, a)$ in replay buffer $\boldsymbol{D}$
    Overwrite the oldest tuple if $|\boldsymbol{D}| > \boldsymbol{D}_{\max}$
    if $len(\boldsymbol{D}) > min_batch$ **then**
      Sample min_batch of α_b randomly.
      Update θ_k^{λ} by using (24).
      Update θ^{Q} using (26).
      Update the parameters of the target network using (27) at every α_f step
    end if
  end for
end for

Table 1
Parameter settings.

Name	Value	Name	Value
C_1	5	C_2	50
C_3	10	C_4	10
β	$1.00 \rightarrow 0.05$	δ	0.95
d_τ	5 m	$v_{\max}$	2.83
d_c	0.5 m	$\boldsymbol{D}_{\max}$	100000
d_o	2 m	α_f	10
α_t	200	α_b	32
c_a^W	0.67	c_o^W	0.5
μ^Q	0.01	μ^{λ}	0.005

4. Simulations and analysis

In this section, we first introduce the parameter settings and evaluation index used in the simulation. Subsequently, a series of numerical simulations were conducted to verify the effectiveness of PGAT-MAPFL algorithm, and the results were compared with those of the traditional MARL algorithm and other traditional algorithms.

4.1. Parameter settings

The PGAT-MAPFL algorithm primarily comprises two networks: actors and critics. For criticism, we adopt a bilayer MLPS containing 64 and 1 neurons. The actor network consists of a PGAT structure in which the fully connected layers of the agent, reference point, and neighbor have 64, 64, and 128 neurons, respectively, and their output values are spliced into MLP containing 64 and 2 neurons. The Adam optimizer was employed to optimize all network parameters, and the proposed algorithm underwent training using a total of 10,000 sets within a randomly initialized system state. During training, σ decayed exponentially from an initial value of 1.0 to a minimum value of 0.05. Due to the motion limitation of the minimum turning radius of each swarm, the ideal spacing between each group of swarms was 2 m. The goal of the mission is to make the UAV group complete the movement from one area to another in the plane, in which the UAV will not collide with other swarms or obstacles in the group, and maintain the ideal formation in the final target area. Table 1 shows the empirical values for parameter Settings.

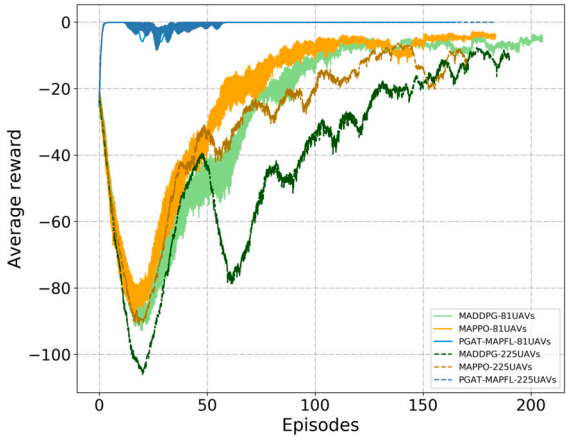

Fig. 7. Learning curves of MADDPG, MAPPO, and PGAT-MAPFL for different numbers of UAVs in swarm.

4.2. Algorithm evaluation metrics

To quantitatively evaluate the efficiency and results of different algorithms, we define the following performance indicators in detail to measure the results:

- **Average reward** $\overline{R}_\tau$: In order to evaluate the performance of the algorithm during the training phase, the average reward of a certain number of episodes α_e is defined:

$$\overline{R}_\tau = \frac{1}{k_\tau \alpha_e \alpha_t} \sum_{h=1}^{h_\tau} \sum_{z=1}^{\alpha_e} \sum_{t=1}^{\alpha_t} R \tag{28}$$

  where $k_\tau = b_1 b_2 b_3 b_4$ denotes the number of UAVs, and
- **Collision rate** I: When the distance between the swarm and other swarms or obstacles is less than d_c, the collision rate of swarm motion can be defined as

$$I = \frac{1}{k_\tau \alpha_e \alpha_t} \sum_{k=1}^{k_\tau} \sum_{z=1}^{\alpha_e} \sum_{t=1}^{\alpha_t} \mathbb{1}(\|\boldsymbol{p}_{k,l}\| < d_c \text{ or } \|\boldsymbol{p}_{k,o}\| < d_c) \tag{29}$$

  where $\mathbb{1}(\cdot)$ is the indicator function.
- **Average position error** $\overline{e}_p$: The average position error between UAVs and their desired position in the algorithm evaluation stage after training can be expressed as

$$\overline{e}_p = \frac{1}{k_\tau \alpha_t} \sum_{k=1}^{k_\tau} \sum_{t=1}^{\alpha_t} \|\boldsymbol{p}_k - \boldsymbol{p}_{k'}\| \tag{30}$$

  where $\boldsymbol{p}_{k'}$ is the desired position of the agent k.
- **Average yaw angle error** $\overline{e}_\psi$: The average yaw angle error between UAVs and their reference points can be expressed as

$$\overline{e}_\psi = \frac{1}{k_\tau \alpha_t} \sum_{k=1}^{k_\tau} \sum_{t=1}^{\alpha_t} \psi_k - \psi_{r_k} \tag{31}$$

  where ψ_{r_k} denotes the yaw angle of the reference point of agent k;
- **Average ground speed error** $\overline{e}_v$: The average ground speed error between UAVs and their reference points can be expressed as

$$\overline{e}_v = \frac{1}{k_\tau \alpha_t} \sum_{k=1}^{k_\tau} \sum_{t=1}^{\alpha_t} v_k - v_{r_k} \tag{32}$$

  where v_{r_k} denotes the yaw angle of the reference point of agent k;
- **Group convergence time after passing through the obstacle area** T_c: The speed at which the system reaches the lowest energy point is positively related to the convergence time of the swarm. At the lowest energy point, the swarm can move stably during the set formation.

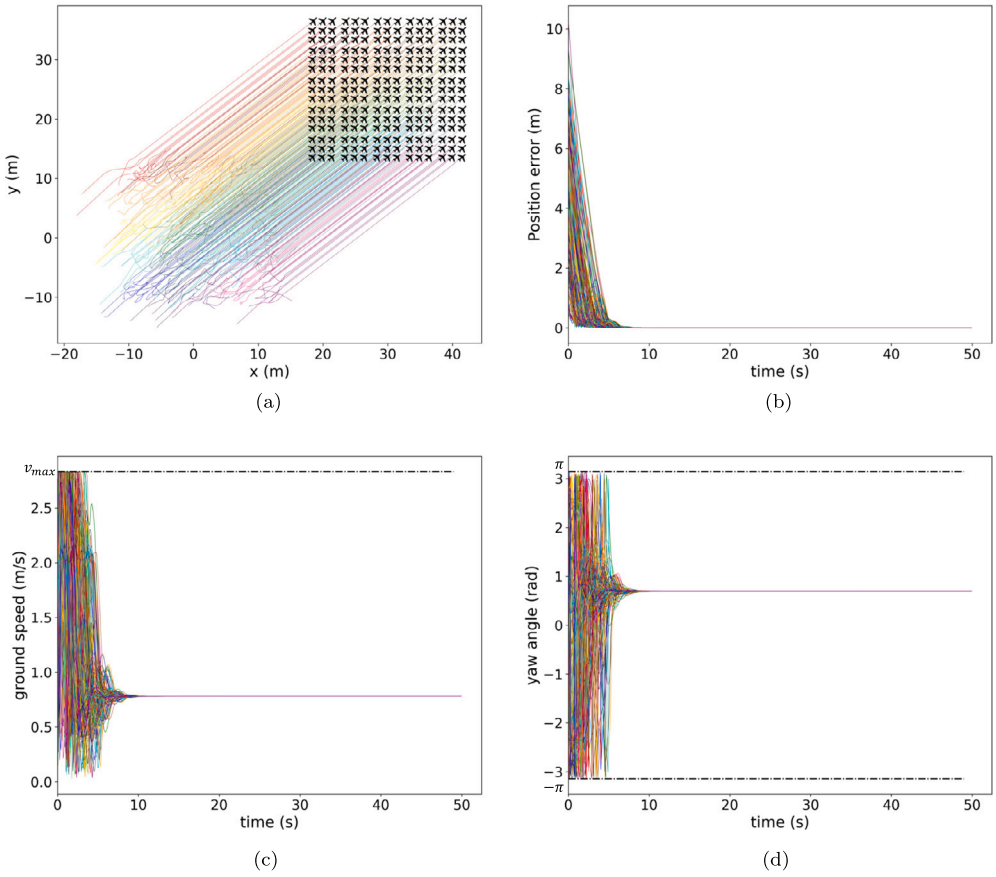

Fig. 8. Simulation results of linear motion of UAVs swarm without obstacles. (a) Trajectory of UAVs. (b) Position error of UAVs. (c) Ground speed of UAVs. (d) Yaw angle of UAVs.

- **The number of UAVs with the same speed after the system converges** k_s. Ideally, all drones in the cluster will have the same speed after convergence.
- **Program runtime** T_p: When using the same hardware device and simulation environment, the shorter the program running time, the lower the computational complexity. This index is used to confirm that the calculation is suitable for large-scale UAV clusters. The hardware was equipped with an AMD Ryzen 7 5800H with Radeon Graphics 3.20 GHz CPU, 16G memory, and 512G solid state disk.
- **Average flight path length of UAVs** $\overline{d}_f$: The shorter the path, the smoother the trajectory, which means the lower the energy cost required.

4.3. Training results

During training, we compared PGAT-MAPFL to the baseline MADDPG [51] and MAPPO [53] with different numbers of UAVs. $\overline{R}_\tau$ values of the MARL algorithm and PGAT-MAPFL were recorded in the obstacle avoidance training process for different numbers of UAV swarms, and the curves were drawn, as shown in Fig. 7. For the MARL algorithm, in the training process, owing to the early action exploration process, the actions taken by the agent have strong randomness, and no appropriate action strategy is obtained. Thus, the $\overline{R}_\tau$ value has a large negative value. With the continuous optimization of the training strategy, the value of $\overline{R}_\tau$ gradually increases and the algorithm converges. As can be seen from Fig. 7, compared to MADDPG algorithm, PGAT-MAPFL has a faster convergence speed, which indicates that the graph attention mechanism and improved actor network can reduce unnecessary exploration, increase the stability of actions, and accelerate the convergence of the algorithm. In addition, compared with MADDPG, $\overline{R}_\tau$ of PGAT-MAPFL algorithm almost converges to zero, and the curve fluctuation is smaller after convergence. During training, PGAT-MAPFL did not fall into local optimality.

4.4. Testing results and analysis

In this section, the validity of PGAT-MAPFL algorithm in MCOAL is verified by simulations. Consider a UAV swarm of 225 UAVs divided into 25 groups, where sub-matrices B_{ij} are clusters of nine UAVs, that is, $b_1 = 5$, $b_2 = 5$, $b_3 = 3$, $b_4 = 3$, and we verify the performance of the proposed algorithm under different working conditions. Note that all subsequent simulations were conducted in a random initial state, and the same color was used to represent the change curve of the UAVs within the same sub-matrix B_{ij}; the black dots represent obstacles.

4.4.1. Motion control without obstacles

First, the stability and consistency of PGAT-MAPFL in a straight flight without obstacles were verified. We tested both linear and circular motions, and the simulation results are shown in Fig. 8 and 9. Fig. 8(a) and Fig. 9(a) show the trajectory of UAVs, combined with Fig. 8(b) and Fig. 9(b), which show that under the condition of random initial position and speed, UAV can quickly reach the desired position and follow the reference point. The error from the desired position eventually converges to zero; that is, the group can eventually converge to the desired structure, and UAVs can maintain the desired distance during flight. Fig. 8(c) shows the ground speed of all UAVs in the swarm in linear motion, it can be seen that the speed of 225 UAVs can quickly converge to the speed of reference point $v_r = 0.78$ m/s and the maximum speed does not exceed $v_{max} = 2.84$ m/s. Fig. 9(c) shows the ground speed of all UAVs in the swarm in linear motion, where the speed of the reference point $v_r = 1$ m/s. Fig. 8(d) and 9(d) show the variations in the heading Angle of the UAV in a swarm. When moving in straight

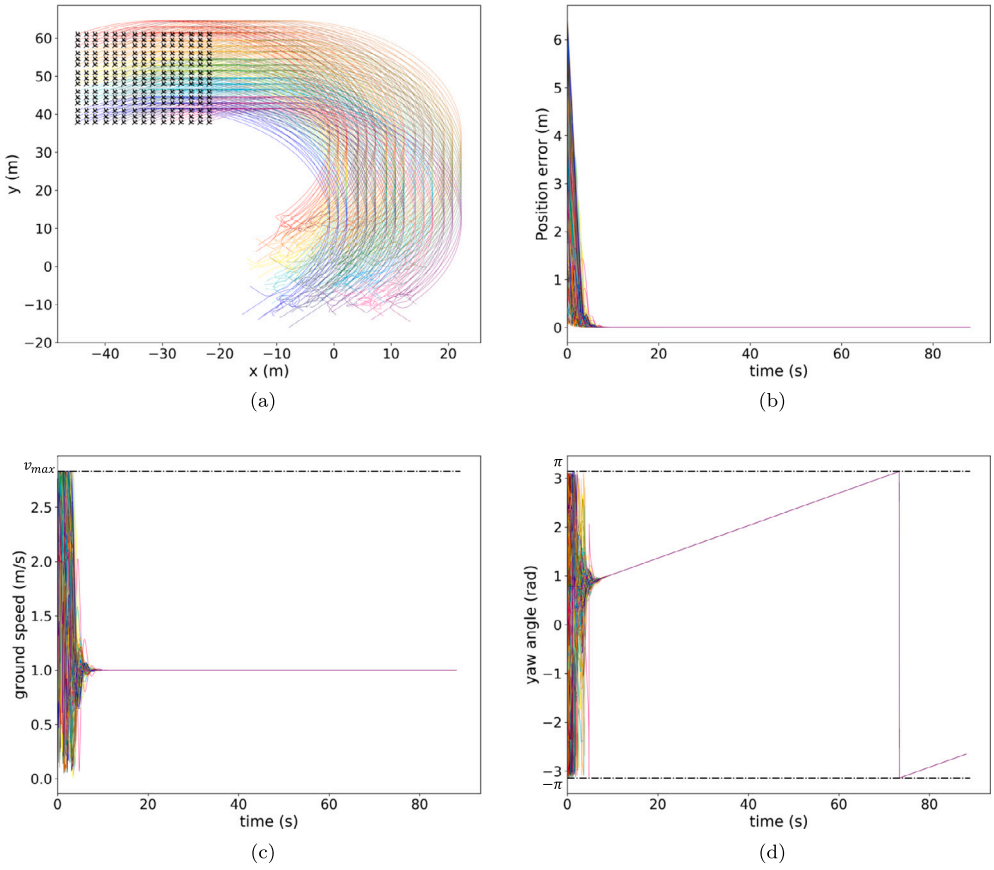

Fig. 9. Simulation results of circular motion of UAVs swarm without obstacles. (a) Trajectory of UAVs. (b) Position error of UAVs. (c) Ground speed of UAVs. (d) Yaw angle of UAVs.

lines, the yaw angles of the UAVs remain the same after convergence. In circular motion, the yaw angle of the UAVs changes with time after convergence, and the error converges to zero. In other words, the proposed algorithm performed well in terms of speed convergence and consistency under collision-free motion. This also implies that the UAVs within the swarm can maintain the required distance during movement, thereby maintaining their formation.

4.4.2. Motion control with static obstacles

To verify the effectiveness of the algorithm proposed in this paper in the obstacle avoidance algorithm of large-scale UAV clusters, we conducted simulations on two scenarios of linear motion and circular motion, the results of which are shown in Fig. 11 and 11. Fig. 10(a) and Fig. 10(a) show the trajectory of UAVs, which can avoid obstacles smoothly, similar to water flowing over a reef. Since there may be multiple obstacles in other nodes connected to the UAV in the graph structure, UAVs can smoothly pass through the obstacle area. Through feature extraction and potential function training of PGAT, the agent makes a series of obstacle avoidance decisions. In addition, the trajectories of UAVs are very similar after avoiding obstacles, which indicates that the proposed algorithm still has stable convergence after avoiding obstacles. Fig. 10(b) and Fig. 11(b) show the position error between UAVs and desired position. It can be observed that after passing through the static obstacle environment, the UAV was always able to stably reach the desired position, even when the reference point was constantly changing. Additionally, it can be seen from Fig. 10(c) and Fig. 11(c) that in order to avoid obstacles and avoid collisions with other UAVs in the cluster, the speed of the UAVs in the area near the obstacles will constantly change, and will quickly converge to the speed of reference point $v_r = 1m/s$ and maintain consistency after moving away from the obstacles. Fig. 10(d) and Fig. 11(d) shows the change of yaw angle in linear and circular motion under an obstacle environment. When approaching an obstacle, the heading angle varies between $-\pi$ and π owing to the need to reverse the direction of motion to avoid obstacles and neighbor UAVs.

The above results demonstrate that the proposed algorithm can reach a destination with the desired structure without UAV collisions; thus, convergence of the algorithm is confirmed. When $t \to \infty$, E_u, E_c, E_m are at their lowest, implying that the energy of the entire system remains in its lowest state.

4.4.3. Motion control with dynamic obstacles

To further assess the efficacy of the proposed obstacle avoidance algorithm for large-scale UAV swarms, we conducted simulations under dynamic obstacle scenarios. The simulation results are depicted in Fig. 12, in which 8 obstacles were considered, each with specific initial positions and velocities. Fig. 12(a) illustrates the flight path of UAVs and obstacles, in which the black dot represents the initial position of the obstacle, the red arrow indicates its direction and the black line traces its trajectory. To enhance visibility of the obstacle avoidance process and facilitate comparison with the static obstacle avoidance scenario, one out of the eight obstacles remains stationary, aligning with the static scene. Given the dynamic movement of obstacles, the UAV must promptly execute obstacle avoidance maneuvers based on obstacle positions. This trend is evident in the magnified portion of Fig. 12(a), demonstrating the effectiveness of PGAT-MAPFL in dynamic environments. Even when obstacles are in motion, the agent can make real-time obstacle avoidance decisions based on locally observed states. Analogous to the static obstacle avoidance scenario, UAVs has the capability to alter its movement direction and circumnavigate obstacles when they come within a specific range. In the conducted simulation, UAVs swarm successfully avoided collisions with obstacles, underscor-

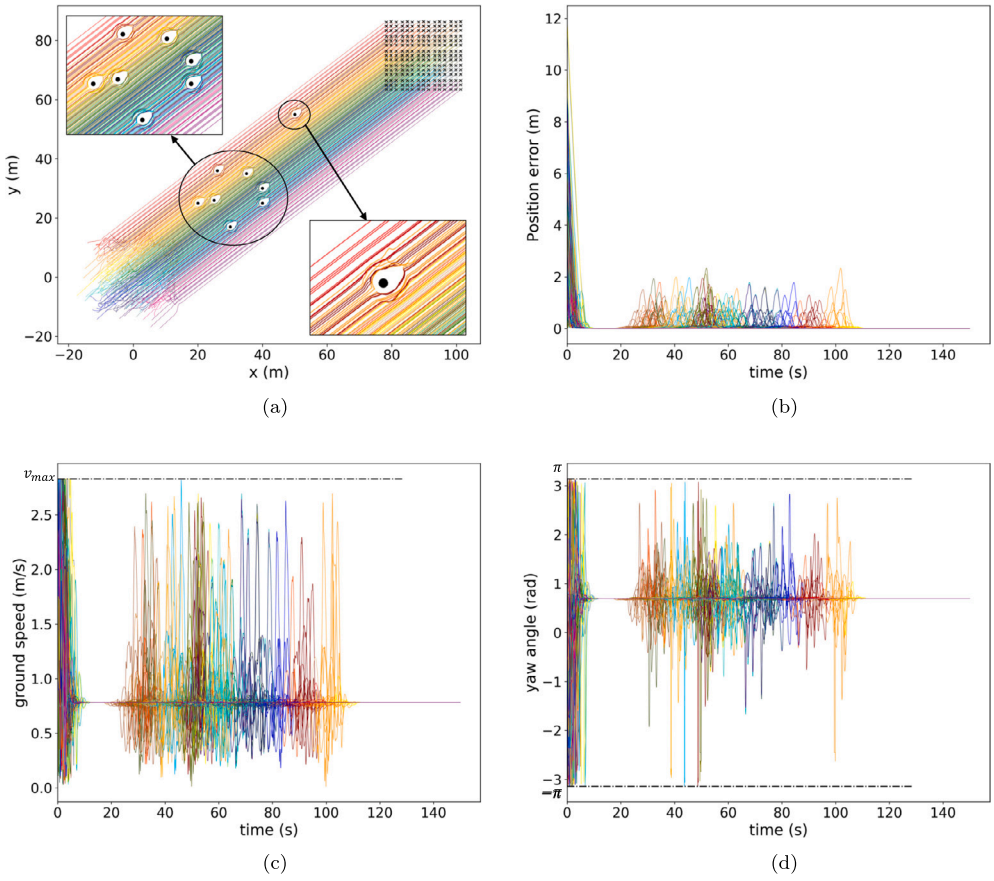

Fig. 10. Simulation results of linear motion of UAVs swarm with static obstacles. (a) Trajectory of UAVs. (b) Position error of UAVs. (c) Ground speed of UAVs. (d) Yaw angle of UAVs.

ing the robust obstacle avoidance efficacy of PGAT-MAPFL even in dynamic obstacle scenarios. Furthermore, the trajectory of the UAV remains remarkably consistent after passing through obstacle areas, indicating stable convergence following obstacle avoidance. Fig. 12(c), Fig. 12(d) and Fig. 12(d) depict the error between the UAV's actual state and desired position, speed, and heading angle, respectively. Notably, the UAV group achieves stable positioning and converges to similar speeds and directions after navigating the dynamic obstacle environment.

4.5. Comparison and analysis

To verify the superiority of the proposed method, we compared the performance of the traditional potential field function method and MADDPG algorithm in MCOAL problem of linear motion, as shown in Fig. 13. It is difficult for model-free actor networks to converge to the theoretical optimal action under different state inputs through strategy gradient descent, although in ideal conditions. As a result, the control accuracy of the MARL algorithm is inferior to that of the traditional algorithm. Meanwhile, there must be a deviation between the actor network output and its theoretical value. In the traditional algorithm, the flight path of a UAV group after passing through an obstacle area is bumpy and expensive, and it takes a long time to restore the formation structure. During these processes, collisions between UAVs may occur.

More detailed results are presented in Table 2. It can be observed that the average error and collision rate of PGAT-MAPFL are smaller than those of the existing algorithms. The trajectory smoothness is better and the convergence is faster after passing through the obstacle area, which means that the proposed algorithm can recover the required formation more quickly at a lower cost. In summary, it can be concluded that the proposed algorithm exhibits a greater performance improvement than existing methods.

Table 2
Comparison of simulation results.

Indicator	PGAT-MAPFL	MADDPG	Method in [37]
I	0	25%	3%
$\overline{e}_p$, m	6.78×10^{-2}	2.69	15.02×10^{-2}
$\overline{e}_\psi$, rad	-1.29×10^{-2}	9.19×10^{-2}	7.34×10^{-2}
$\overline{e}_v$, m/s	2.84×10^{-2}	1.86	2.57×10^{-2}
T_c, s	3.6	NaN[1]	4.4
k_s	225	92	211
T_p, s	12.2	10.1	15.6
$\overline{d}_f$	114	155	123

[1] NAN refers to the fact that the algorithm does not converge completely at $t \to \infty$.

Owing to the response speed of the neural network, the time consumption of each agent from constructing an obstacle group state based on the detected obstacle and neighbor information to the output control quantity through its actor network is approximately 2.23 ms, which means that PGAT-MAPFL has great potential to meet real-time control requirements in an actual dynamic obstacle environment. It can converge stably even in unlearned cluster scenarios, indicating that our algorithm has good adaptability. The main reason is that PGAT can effectively extract important information from local neighbors and reduce the interference of non-neighbor information. As a cooperative learning method, PGAT-MAPFL enables each node to obtain the behavior of its neighbors during the experiential learning process to learn the global optimal behavior strategy for the entire control process. It is worth noting that even if the number of UAVs increased or decreased, the collision rates of PGAT-MAPFL are always less than 1‰. Thus, the per-

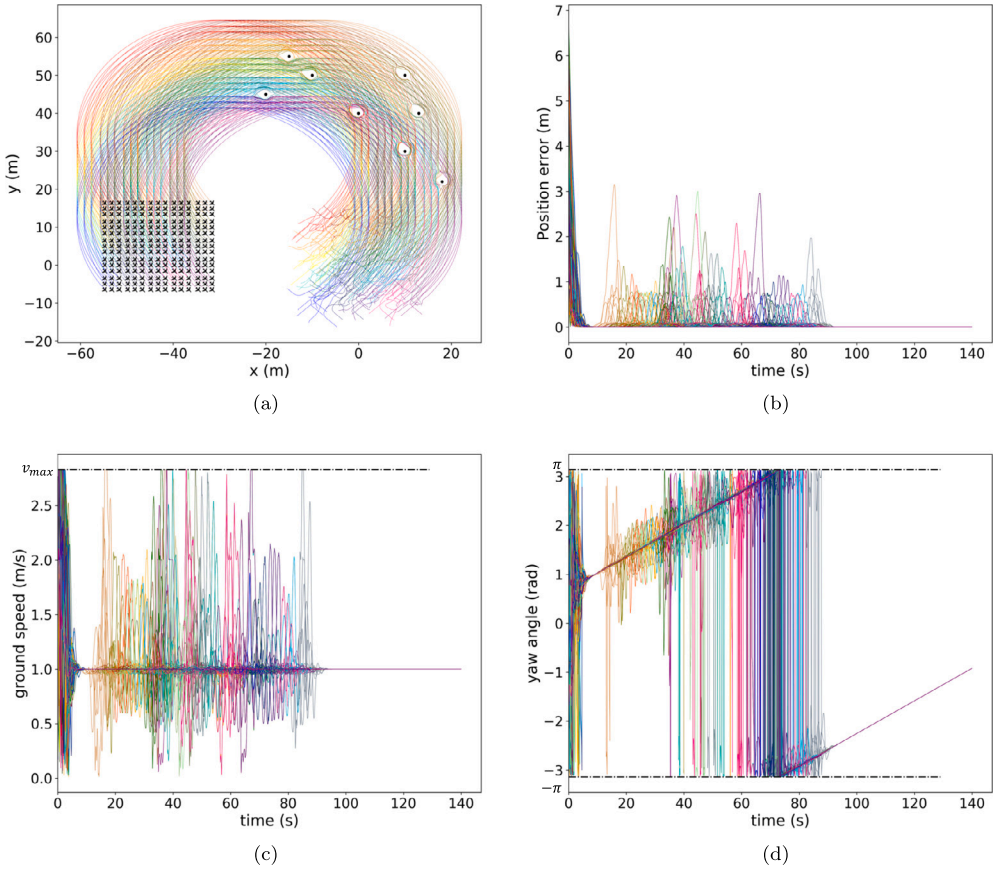

Fig. 11. Simulation results of circular motion of UAVs swarm with static obstacles. (a) Trajectory of UAVs. (b) Position error of UAVs. (c) Ground speed of UAVs. (d) Yaw angle of UAVs.

formance of PGAT-MAPFL is likely to be satisfactory and independent of the population size; that is, the proposed algorithm can be generalized to different population sizes. Although PGAT-MAPFL demonstrates superior performance, it still exhibits limitations for UAVs due to the exclusive consideration of a simplified dynamic model and the prerequisite for pre-deployment training. In our future research endeavors, we intend to explore more intricate dynamic model and investigate training methodologies within real-world contexts to enhance alignment with practical scenarios.

5. Conclusion

A PGAT-MAPFL algorithm based on local position information and a model reference was proposed to solve the problem of real-time control of cooperative movement and obstacle avoidance in large-scale UAV clusters. First, a swarm structure suitable for large-scale UAV motion control was proposed to reserve maneuvering space for UAV during flight. In addition, a position-based graph attention mechanism is proposed that enables each agent to output actions according to local information to achieve distributed control. The designed PGAT enables the UAV to obtain neighbor information, improves the agent's ability to understand the observed and system states, and reduces the influence of messy information in the observed state on the control strategy. Second, the entire cluster system adopts the same evaluation network in the training, and the strategy evaluation mechanism based on global reward considers system loss as the strategy optimization goal to achieve cooperative control with minimum system loss. Similar to model-reference reinforcement learning, we introduce a potential field function into the actor network to output the control quantity and learn its magnitude to achieve collision avoidance, obstacle avoidance, and rapid formation reconstruction. Simulation results show that, compared with other RL-based algorithms and traditional methods, PGAT-MAPFL algorithm has the best control stability, minimum system control error, faster convergence speed, better learning effect, and less computing resource consumption. Simultaneously, the PGAT-MAPFL algorithm has the scalability of different cluster sizes and adaptability of real-time distributed control. In conclusion, the proposed algorithm provides a more effective method for future research on motion and obstacle avoidance in large-scale UAV swarm systems.

CRediT authorship contribution statement

Bocheng Zhao: Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Mingying Huo:** Investigation, Formal analysis, Data curation. **Zheng Li:** Validation, Supervision, Resources, Conceptualization. **Ze Yu:** Validation, Software, Resources, Investigation, Funding acquisition. **Naiming Qi:** Resources, Project administration, Methodology, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

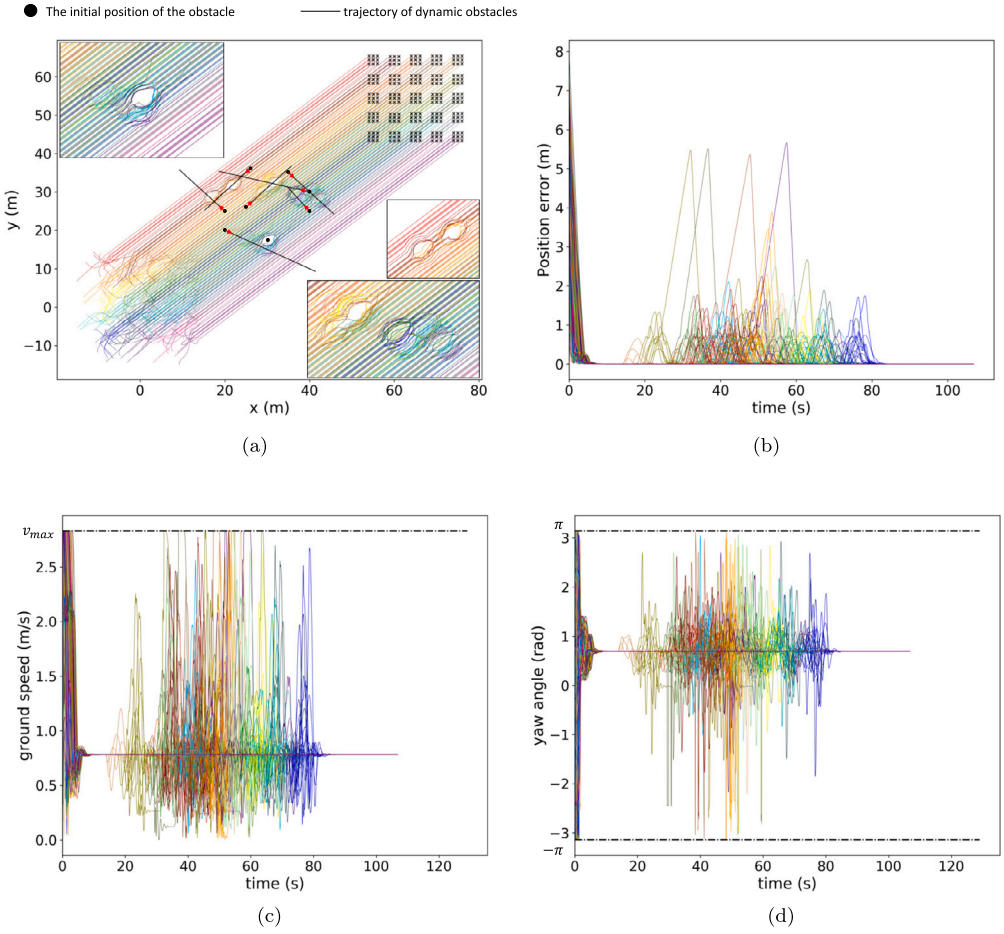

Fig. 12. Simulation results of linear motion of UAVs swarm with dynamic obstacles. (a) Trajectory of UAVs. (b) Position error of UAVs. (c) Ground speed of UAVs. (d) Yaw angle of UAVs.

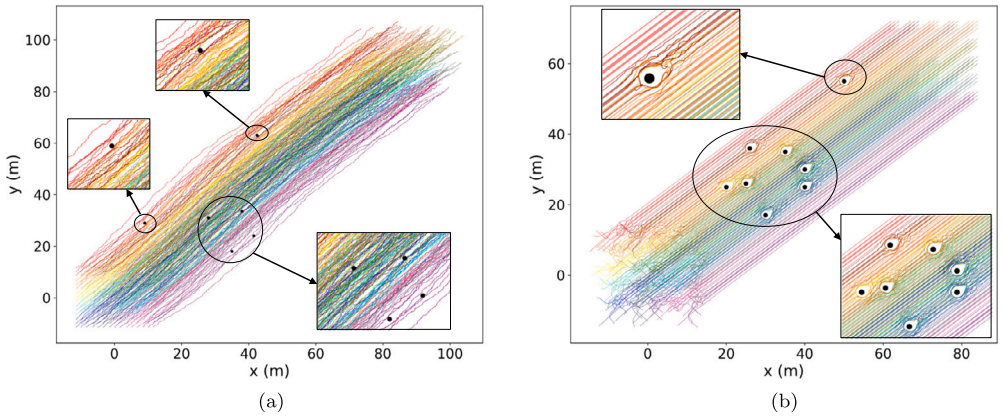

Fig. 13. Simulation results of other methods. (a) Trajectory of MADDPG. (b) Trajectory of traditional methods.

References

[1] X. Shao, Y. Xia, Z. Mei, W. Zhang, Model-guided reinforcement learning enclosing for uavs with collision-free and reinforced tracking capability, Aerosp. Sci. Technol. 142 (2023) 108609.

[2] X. Liu, D. Zhao, N.L. Oo, Comparison studies on aerodynamic performances of a rotating propeller for small-size uavs, Aerosp. Sci. Technol. 133 (2023) 108148.

[3] Y. Zhang, S. Li, S. Wang, X. Wang, H. Duan, Distributed bearing-based formation maneuver control of fixed-wing uavs by finite-time orientation estimation, Aerosp. Sci. Technol. 136 (2023) 108241.

[4] X. Shao, H. Liu, W. Zhang, J. Zhao, Q. Zhang, Path driven formation-containment control of multiple uavs: a path-following framework, Aerosp. Sci. Technol. 135 (2023) 108168.

[5] M. Zhang, C. Liang, J. Mei, Robust guidance law for cooperative aerial target circumnavigation of uavs based on composite system theory, Aerosp. Sci. Technol. (2023) 108439.

[6] L. Wen, Z. Zhen, T. Wan, Z. Hu, C. Yan, Distributed cooperative fencing scheme for uav swarm based on self-organized behaviors, Aerosp. Sci. Technol. 138 (2023) 108327.

[7] B. Zhao, M. Huo, Z. Yu, N. Qi, J. Wang, Model-reference reinforcement learning for safe aerial recovery of unmanned aerial vehicles, Aerospace 11 (1) (2023) 27.

[8] Y. Li, L. Zhang, B. Cai, Y. Liang, Unified path planning for composite uavs via Fermat point-based grouping particle swarm optimization, Aerosp. Sci. Technol. (2024) 109088.

[9] H. Zhao, Y. Wen, S. Wu, J. Deng, Dynamic evaluation strategies for multiple aircrafts formation using collision and matching probabilities, IEEE/CAA J. Autom. Sin. 8 (4) (2020) 890–904.

[10] C. Bao, P. Wang, R. He, G. Tang, Observer-based optimal control method combination with event-triggered strategy for hypersonic morphing vehicle, Aerosp. Sci. Technol. 136 (2023) 108219.
[11] Q. Li, D. Gao, C. Sun, S. Song, Z. Niu, Y. Yang, Prescribed performance-based robust inverse optimal control for spacecraft proximity operations with safety concern, Aerosp. Sci. Technol. 136 (2023) 108229.
[12] L. Cheng, H. Wen, J. Kang, D. Jin, Dynamic tube model predictive control for powered-descent guidance, J. Aerosp. Eng. 35 (6) (2022) 04022098.
[13] H. Nguyen, H.B. Dang, P.N. Dao, On-policy and off-policy q-learning strategies for spacecraft systems: an approach for time-varying discrete-time without controllability assumption of augmented system, Aerosp. Sci. Technol. (2024) 108972.
[14] B. Zhang, J. Guo, H. Wang, S. Tang, Autonomous morphing strategy for a long-range aircraft using reinforcement learning, Aerosp. Sci. Technol. (2024) 109087.
[15] M. Liu, H. Zhang, J. Yang, T. Zhang, C. Zhang, L. Bo, A path planning algorithm for three-dimensional collision avoidance based on potential field and b-spline boundary curve, Aerosp. Sci. Technol. 144 (2024) 108763.
[16] W. Zhang, H. Wen, Motion planning of a free-flying space robot system under end effector task constraints, Acta Astronaut. 199 (2022) 195–205.
[17] X. Ma, L. Huang, H. Wen, S. Xu, Deep learning-based nonlinear model predictive control of the attitude manoeuvre of a barbell electric sail through voltage regulation, Acta Astronaut. 195 (2022) 118–128.
[18] Z. Wei, H. Wen, H. Hu, D. Jin, Ground experiment on rendezvous and docking with a spinning target using multistage control strategy, Aerosp. Sci. Technol. 104 (2020) 105967.
[19] D. Jiang, Z. Cai, Z. Liu, H. Peng, Z. Wu, An integrated tracking control approach based on reinforcement learning for a continuum robot in space capture missions, J. Aerosp. Eng. 35 (5) (2022) 04022065.
[20] Z. Wei, T. Chen, H. Wen, D. Jin, H. Hu, Experimental study on autonomous assembly of multiple spacecraft simulators in a spinning scenario, Acta Astronaut. 207 (2023) 106–117.
[21] S. Huang, H. Zhang, Z. Huang, Multi-uav collision avoidance using multi-agent reinforcement learning with counterfactual credit assignment, arXiv preprint, arXiv:2204.08594, 2022.
[22] J. Li, Y. Fang, H. Cheng, Z. Wang, Z. Wu, M. Zeng, Large-scale fixed-wing uav swarm system control with collision avoidance and formation maneuver, IEEE Syst. J. 17 (1) (2022) 744–755.
[23] N.K. Long, K. Sammut, D. Sgarioto, M. Garratt, H.A. Abbass, A comprehensive review of shepherding as a bio-inspired swarm-robotics guidance approach, IEEE Trans. Emerg. Top. Comput. Intell. 4 (4) (2020) 523–537.
[24] S.-J. Chung, A.A. Paranjape, P. Dames, S. Shen, V. Kumar, A survey on aerial swarm robotics, IEEE Trans. Robot. 34 (4) (2018) 837–855.
[25] S. Chen, H. Pei, Q. Lai, H. Yan, Multitarget tracking control for coupled heterogeneous inertial agents systems based on flocking behavior, IEEE Trans. Syst. Man Cybern. Syst. 49 (12) (2018) 2605–2611.
[26] X. Shao, Z. Cao, H. Si, Neurodynamic formation maneuvering control with modified prescribed performances for networked uncertain quadrotors, IEEE Syst. J. 15 (4) (2020) 5255–5266.
[27] G. Jing, L. Wang, Multiagent flocking with angle-based formation shape control, IEEE Trans. Autom. Control 65 (2) (2019) 817–823.
[28] M. Waibel, B. Keays, F. Augugliaro, Drone shows: creative potential and best practices, Tech. Rep., ETH, Zurich, 2017.
[29] X. Zheng, C. Zong, J. Cheng, J. Xu, S. Xin, C. Tu, S. Chen, W. Wang, Visually smooth multi-uav formation transformation, Graph. Models 116 (2021) 101111.
[30] D. Jiang, Z. Cai, H. Peng, Z. Wu, Coordinated control based on reinforcement learning for dual-arm continuum manipulators in space capture missions, J. Aerosp. Eng. 34 (6) (2021) 04021087.
[31] Z. Wenhong, L. Jie, L. Zhihong, S. Lincheng, Improving multi-target cooperative tracking guidance for uav swarms using multi-agent reinforcement learning, Chin. J. Aeronaut. 35 (7) (2022) 100–112.
[32] X. Wang, Y. Wang, X. Su, L. Wang, C. Lu, H. Peng, J. Liu, Deep reinforcement learning-based air combat maneuver decision-making: literature review, implementation tutorial and future direction, Artif. Intell. Rev. 57 (1) (2024) 1.
[33] B. Chen, B. Dai, L. Song, Learning to plan via neural exploration-exploitation trees, arXiv preprint, arXiv:1903.00070, 2019.
[34] C.K. Joshi, T. Laurent, X. Bresson, An efficient graph convolutional network technique for the travelling salesman problem, arXiv preprint, arXiv:1906.01227, 2019.
[35] E. Tolstaya, J. Paulos, V. Kumar, A. Ribeiro, Multi-robot coverage and exploration using spatial graph neural networks, in: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, IEEE, 2021, pp. 8944–8950.
[36] J. Paulos, S.W. Chen, D. Shishika, V. Kumar, Decentralization of multiagent policies by learning what to communicate, in: 2019 International Conference on Robotics and Automation, ICRA, IEEE, 2019, pp. 7990–7996.
[37] T.Z. Muslimov, R.A. Munasypov, Consensus-based cooperative control of parallel fixed-wing uav formations via adaptive backstepping, Aerosp. Sci. Technol. 109 (2021) 106416.
[38] C. Yan, X. Xiang, C. Wang, F. Li, X. Wang, X. Xu, L. Shen, Pascal: population-specific curriculum-based madrl for collision-free flocking with large-scale fixed-wing uav swarms, Aerosp. Sci. Technol. 133 (2023) 108091.
[39] F. Chen, S. Bai, T. Shan, B. Englot, Self-learning exploration and mapping for mobile robots via deep reinforcement learning, 2019.
[40] F.A. Oliehoek, C. Amato, et al., A Concise Introduction to Decentralized POMDPs, vol. 1, Springer, 2016.
[41] J. Xiao, G. Yuan, Z. Wang, A multi-agent flocking collaborative control method for stochastic dynamic environment via graph attention autoencoder based reinforcement learning, Neurocomputing (2023) 126379.
[42] L. Wei-heng, Z. Xin, D. Zhi-hong, Dynamic collision avoidance for cooperative fixed-wing uav swarm based on normalized artificial potential field optimization, J. Cent. South Univ. 28 (10) (2021) 3159–3172.
[43] C. Schroeder de Witt, J. Foerster, G. Farquhar, P. Torr, W. Boehmer, S. Whiteson, Multi-agent common knowledge reinforcement learning, Adv. Neural Inf. Process. Syst. 32 (2019).
[44] M. Schlichtkrull, T.N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, M. Welling, Modeling relational data with graph convolutional networks, in: The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings 15, Springer, 2018, pp. 593–607.
[45] D. Liben-Nowell, J. Kleinberg, The link prediction problem for social networks, in: Proceedings of the Twelfth International Conference on Information and Knowledge Management, 2003, pp. 556–559.
[46] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, Graph attention networks, arXiv preprint, arXiv:1710.10903, 2017.
[47] W. Zhang, G. Xu, Y. Song, Y. Wang, An obstacle avoidance strategy for complex obstacles based on artificial potential field method, J. Field Robot. 40 (5) (2023) 1231–1244.
[48] X. Tong, S. Yu, G. Liu, X. Niu, C. Xia, J. Chen, Z. Yang, Y. Sun, A hybrid formation path planning based on a* and multi-target improved artificial potential field algorithm in the 2d random environments, Adv. Eng. Inform. 54 (2022) 101755.
[49] J.-J. Chen, H.-C. Hung, Y.-R. Sun, J.-H. Chuang, Apf-s2t: steering to target redirection walking based on artificial potential fields, IEEE Trans. Vis. Comput. Graph. (2024).
[50] J. Xiao, Z. Wang, J. He, G. Yuan, A graph neural network based deep reinforcement learning algorithm for multi-agent leader-follower flocking, Inf. Sci. 641 (2023) 119074.
[51] T.P. Lillicrap, J.J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, D. Wierstra, Continuous control with deep reinforcement learning, arXiv preprint, arXiv:1509.02971, 2015.
[52] R. Lowe, Y.I. Wu, A. Tamar, J. Harb, O. Pieter Abbeel, I. Mordatch, Multi-agent actor-critic for mixed cooperative-competitive environments, Adv. Neural Inf. Process. Syst. 30 (2017).
[53] O. Lohse, N. Pütz, K. Hörmann, Implementing an online scheduling approach for production with multi agent proximal policy optimization (mappo), in: Advances in Production Management Systems. Artificial Intelligence for Sustainable and Resilient Production Systems: IFIP WG 5.7 International Conference, APMS 2021, Nantes, France, September 5–9, 2021, Proceedings, Part V, Springer, 2021, pp. 586–595.