Multi-UAV Navigation for Partially Observable Communication Coverage by Graph Reinforcement Learning

Zhenhui YE, Ke WANG, Yining CHEN, Xiaohong JIANG, Guanghua SONG

Abstract—In this paper, we aim to design a deep reinforcement learning(DRL) based control solution to navigate a swarm of unmanned aerial vehicles (UAVs) to fly around an unexplored target area under partial observation, serving as mobile Base Stations (BSs) to provide optimal communication coverage for the ground mobile users. Compared with existing DRL-based solutions that mainly solve the problem with global observation and centralized training, a practical and efficient Decentralized Training and Decentralized Execution(DTDE) framework is desirable to train and deploy each UAV in a distributed manner. To this end, we propose a novel DRL approach named Deep Recurrent Graph Network(DRGN) that makes use of Graph Attention Network-based Flying Ad-hoc Network(GAT-FANET) to achieve inter-UAV communications and Gated Recurrent Unit (GRU) to record historical information. We conducted extensive experiments to define an appropriate structure for GAT-FANET and examine the performance of DRGN. The simulation results show that the proposed model outperforms four state-of-the-art DRL-based approaches and four heuristic baselines, and demonstrate the scalability, transferability, robustness, and interpretability of DRGN.

Index Terms—UAV Control, Communication Coverage, Deep Reinforcement Learning, Graph Learning, Decentralized Training.

1 INTRODUCTION

WITH the development of advanced technologies such as onboard computers [1], robust flight control stacks [2], and Ad-hoc networks [3], it is feasible to control large-scale intelligent UAV swarms to complete complex tasks. Nowadays, the UAV swarm has been applied in various scenarios such as reconnaissance [4], transportation [5], power inspection [6], disaster relief [7]. Specifically, UAVs mounted mobile base stations(UAV-MBSs) can be deployed to improve the coverage and performance of communication services in a target area without enough existing communication infrastructure [8]. Compared with ground stations, UAV-MBSs could be quickly and dynamically deployed in emergent scenarios such as a catastrophic natural disaster where cellular communication networks are disrupted [9], or providing emergency communications and network access for remote areas [10]. In recent years, UAV-MBSs is generally considered as a cost-efficient communication base station with high *Quality-of-Services*(QoS) [11] [12] [13].

The system modeling of the UAV-MBS problem has been extensively studied by recent works [14] [15]. In previous studies, each UAV is generally equipped with a network device such as WiFi, NB-IoT, or LoRA, which enables it to communicate with other UAVs and provides network access for ground users within the communication range. Due to the constraints of maximum communication distance and high mobility, the UAVs need to cooperate autonomously to provide better communication services for the ground users, as illustrated in Fig 1. The goal of the multi-UAV system is to provide ground users with a high coverage rate and ensure coverage fairness among the users.

Compared with previous works, our modeling of the problem is more practical and challenging. Firstly, instead of assuming that ground users are evenly distributed [16] [17], we assume that the ground users are randomly distributed

Fig. 1. Partially observable multi-UAV navigation scenario.

and the UAV can only observe ground users within the maximum communication distance. This setting requires the policy to have a good robustness and exploration capability. Secondly, since it is costly to obtain global information in real tasks, we assume that the UAV can only obtain information from itself and its interconnected neighbors, while the global information is unavailable during training. As such, we could only use local observations to calculate a local metric to evaluate and update the policy, yet in the testing phase, the decentralized-learned policy needs to achieve high performance in terms of several global metrics. Thirdly, due to the limitations such as energy or communication interference, the actual number of UAVs to be controlled may change, thus we require the collaborative algorithm to be scalable and transferrable.

Our intuition to solve this partially observable problem is to make use of the *flying Ad-hoc network*(FANET), where each pair of UAVs within the maximum communication

distance are interconnected and could communicate with low latency. Taking into account the dynamic topology and graph characteristics of the UAV Ad-hoc network, we regard each UAV as a node and each connection in FANET as an edge. Then we use the *graph attention network*(GAT) [18] structure to aggregate adjacent information through the edges. This communication mechanism between neighbors could alleviate the influence of partial observation at a low cost and is named as *GAT-based FANET*(GAT-FANET). Besides, *gated recurrent unit*(GRU) is adopted to record the significant historical information to handle the partial observation.

The main contributions of this paper are as follows:

- 1) We propose a novel network architecture named *Deep Recurrent Graph Network*(DRGN), which is based on the aforementioned GAT-FANET and GRU. DRGN is a lightweight network so that it could be easily deployed in the onboard computer of the UAV. Besides, as we assume that global information is not available, a reward function that only evaluates the performance of the UAV and its one-hop neighbors is designed to train each UAV distributively. And our experiments demonstrate that this *local* reward function manages to encourage the UAVs to perform well in terms of global metrics including coverage and fairness.
- 2) We conducted extensive simulations to find appropriate structures for the GAT-based Ad-hoc network and analyze several characteristics of DRGN, including performance, scalability, transferability, robustness, as well as interpretability.
- 3) We compared our DRGN model with four state-of-art DRL methods and four heuristic baselines. Results show that our algorithm consistently outperforms the others.

The rest of the paper is organized as follows. In Sec.2, we review recent works related to multi-UAV navigation. The system model and problem statement are defined in Sec.3. Necessary background knowledge for DRGN is given in Sec.4. We describe our approach in Sec.5. Experimental studies, including network architecture search, evaluation, and analysis are given in Sec.6. Finally, we draw conclusions and prospect the future work in Sec.7

2 RELATED WORKS

2.1 UAV deployment and UAV Ad-hoc Network

Some recent works have made in-depth studies on the deployment of UAVs to make it more practical in real-world tasks. The authors in [19] proposed a multi-UAV control model that could increase the deployment coverage of UAVs with energy efficiency. [20] proposed a decentralized solution for multi-UAV deployment by adjusting a set of parameters that control the UAV's behaviors and actions. [21] studied to minimize the deployment delay of all UAVs and the overall delay. [22] proposed a framework for optimizing the deployment and mobility of multiple UAVs to control the overall deployment with regard to the energy efficiency of the ground equipment.

With recent progress in Ad-hoc network [3], where all nodes within the communication range could establish a connection, it is practical to design multi-UAV control methods with the assumption that the UAV could communicate with neighboring UAVs in low-latency. In recent years, Ad-hoc network for multi-UAV deployment, known as *Flying Ad-hoc Network*(FANET), has been proved to have better performance than other network structures [23] [24].

2.2 Cooperative Exploration and Path Planning

A control policy with high exploration efficiency is necessary when the UAV swarm is deployed in an unexplored environment under limited observation. The most widely-used exploration strategy may be ϵ -greedy, which has the probability of ϵ to select actions randomly in each decision, and ϵ often decays with the convergence of the policy. In recent years, many advanced methods have been proposed for multi-UAV cooperative exploration. For instance, [25] developed a game theory framework suitable for multi-UAV collaborative search and monitoring. [26] proposed a cooperative exploration strategy specialized for the UAV-UGV-combined system, which could minimize the total exploration distance under energy consumption and functional constraints. [27] used multi-UAV to build a slam-based collaborative exploration system, which reduces the amount of shared data by only exchanging the frontier points of the computed local grid map.

Before the emergence of DRL, the path planning for multi-UAV is mainly studied with heuristic methods. [28] proposed a multi-objective optimization algorithm for multi-UAV assigning tasks and planning paths, in which *genetic algorithm*(GA) is adopted to minimize the inference time of the policy. [29] designed a variety of path planning algorithms based on the information of each station according to a fixed station, to find a path suitable for the central drone (UAV). [30] proposed the *mean-field game*(MFG) control theory to achieve fast positioning and low flight consumption, in which two partial differential equations are solved by machine learning methods.

2.3 DRL for multi-UAV navigation

Deep reinforcement learning(DRL) is a powerful tool that uses neural networks to learn policies to solve decision-making problems. Since the inference of deep neural networks has linear time complexity, it has better practicability than time-consuming heuristic methods. In recent years, many works have tried to use DRL models to control the multi-UAV navigation in real-world-complexity tasks, and their performance improves with the progress of *multi-agent deep reinforcement learning*(MADRL).

Independent Q-learning(IQL) is probably the simplest and most commonly applied method in the MADRL field [31]. It decomposes the multi-agent problem into multiple simultaneous single-agent tasks, by learning a decentralized Q-learning model for each agent. [32] designed UAV longitudinal and lateral Q-learning fuzzy controllers to solve the multi-UAV formation control problem. [33] handled the problem of the flocking of small fixed-wing UAVs with IQL, where the technique of *parameter sharing* among the

DRL models of all UAVs is applied to speed up the model convergence.

To address the environmental instability caused by training multiple policies simultaneously, [17] introduced the paradigm of *centralized training and decentralized execution* (CTDE), which trains each decentralized policy with a centralized critic network that is granted with a global state of the environment, then executes the policies in a decentralized manner. [15] adopts MADDPG [17] to navigate a group of UAVs as mobile base stations to provide long-term communication coverage for the ground mobile users in a target area. [34] proposed a CTDE MADRL model to provide secure communications by jointly optimizing the trajectory of UAV-MBSs, which applies self-attention mechanism in MADDPG to improve the efficiency of the information aggregation among UAVs.

Note that the CTDE approach requires global information in the training phase, which is normally infeasible in real-world tasks. To achieve fully decentralized training, DGN [35] utilized GAT [18] network to aggregate information from neighboring agents. As is mentioned in Sec. 2.1, with the development of FANET, it is much easier to achieve communications among adjacent UAVs, so it is practical to use GAT structure in the multi-UAV collaboration scenario. Most recently, [36] use the DGN network proposed in [35] to solve the UAV-MBSs problem [15].

The differences between this paper and previous works are obvious. [15] proposes the UAV-MBS problem and solve it with MADDPG, yet it requires full observation during training and evenly distributed points of interest (PoIs); whilst we assume that the model could only obtain a partial observation in the training phase, and the PoIs are randomly distributed. [36] adopts DGN to the UAV-MBS problem, yet the structure of the GAT layers is not well-studied and its performance has no significant advantage over other model-free heuristic methods; whilst we design GAT-FANET, a network structure appropriate for the inter-UAV communication in the UAV-MBSs task, and introduce GRU unit in the network. Further, [14] [15] [36] tested the algorithms with a relatively small UAV team (less than 10); whilst we tested the model in environments with up to 80 UAVs, to validate the scalability of the proposed model.

3 SYSTEM MODEL AND PROBLEM STATEMENT

3.1 System Model

We introduce a scenario of multi-UAV navigation control for fair communication coverage under partial observation. The simulation world is a 2D map evenly divided into $L \times L$ grids, as shown in Fig. 1. A group of N UAVs navigate at a fixed altitude H and serve as mobile communication base stations to provide communication services to ground users. We set D_{Com} as the maximum communication distance within which a UAV can communicate with other UAVs or ground users. Thus, the UAVs are interconnected in the Ad-hoc network when their relative distance is less than D_{Com} . As shown in Fig. 1, a UAV can communicate with the ground users whose distance to the center of UAV on the 2D map is less than R_{Obs} , where $R_{Obs} < D_{Com}$ always holds. Considering real-world constraints on communication quality, we assume that a ground user can obtain a

TABLE 1
List of important notations used in the paper

Notation	Explanation
k, K	PoI index, the number of PoIs
G_i	Subgraph i , which consists of UAV i and its one-hop neighbors
t, T	Timeslot index, the maximum timeslot of an episode
i, N	UAV index, the number of UAV
H	The fixed flight height of each UAV
D_{Com}	Maximum communication distance
R_{Obs}, R_{Cov}	Observation range, coverage range
o_t^i, o_i	Local observation of UAV i at timeslot t
$\mathcal{O}_t^i, \mathcal{O}_i$	Observations of UAVs in G_i at timeslot t
a_t^i, a_i	Action of UAV i at timeslot t
r_t^i, r_i	Reward of UAV i at timeslot t
$\mathbf{o}_t, \mathbf{a}_t, \mathbf{r}_t$	Observations, actions, rewards of all UAVs at timeslot t
$h_t^i, \mathcal{H}_t^i$	Hidden state of UAV's GRU i at timeslot t , hidden states of UAVs in G_i at timeslot t
c_t, c_T	Coverage index at timeslot t , final coverage index at timeslot T
f_t, f_T	Fairness index at timeslot t , final fairness index at timeslot T

stable communication service from a UAV if its distance to the center of the UAV is less than R_{Cov} ; if the distance is more than R_{Cov} and less than R_{Obs} , the ground user can be observed by the UAV and provided with an unstable communication service. We use randomly distributed PoIs to represent the communication demand in the cell.

We consider a task where all UAVs navigate to cover PoIs for T timeslots. The PoIs are randomly distributed in each task and the UAV can only observe the PoI within its observation area. So in order to cover more PoIs, the UAVs should cooperate to explore the distribution of PoIs and choose the trajectory that maximizes the teams' interest in a decentralized manner. At the beginning of each task, the distribution of PoIs is reset and the UAVs take off from random positions, and learn to move to an adjacent cell or simply hover at the current position.

Table 1 lists important notations used in this paper.

3.2 Evaluation Metrics and Problem Statement

We next introduce two important evaluation metrics, then state the goal of this problem. Following [15], we evaluate the model from the perspectives of coverage and fairness. The first is the *Coverage Index*, which measures how every PoI was covered by any UAV in the past t time slots. At any time slot t , if the PoI k is in the coverage area of any UAV, we call it "covered", otherwise it is not covered in t . Specifically, the coverage index in timeslot t can be represented as:

$$c_t = \frac{\sum_{k=1}^K w_t(k)}{Kt} \quad (1)$$

where K is the number of PoI, $w_t(k)$ is the number of timeslot that a PoI k was covered until the timeslot t , and

therefore $c_t \in [0, 1]$ always holds. Since the coverage index at any time t describes the coverage of the PoIs from the beginning of the task to timeslot t , we refer the coverage index at the last timeslot T to *final coverage index*, denoted as $c_T = c_{t|t=T}$, to evaluate the performance of the UAV team in an episode.

However, the final coverage index could be high even when a small subset of PoIs are never covered, therefore, we consider that the fairness of coverage is important. For example, during earthquakes or storms, we hope that even an isolated PoI can have the opportunity to obtain communication services. Besides, as we consider a partially observable environment, we want to evaluate the exploration capability of the policy by verifying whether it could cover the remote/isolated PoIs. Following [14] [15], we use Jain's fairness index to describe the geographical fairness in the coverage of PoIs, as:

$$f_t = \frac{(\sum_{k=1}^K w_t(k))^2}{K \sum_{k=1}^K (w_t(k))^2} \quad (2)$$

Obviously, $f_t \in [\frac{1}{K}, 1]$. Further, if all $w_t(k)$ are equal, f_t is 1; if only one PoI is covered, f_t is $\frac{1}{K}$. We refer the fairness index in the last timeslot T as *final fairness index*, denoted as $f_T = f_{t|t=T}$, to evaluate the coverage fairness and the exploration level in an episode.

In addition to the above metrics, considering the difficulty in acquiring a global state in the real-world task, we add a constraint to the problem that the model will not be granted with the global information in the training phase. It means that the coverage index and fairness index cannot be obtained to formulate the objective function in the training phase. This setting is relevant to the keyword *fully decentralized* in MADRL, which requires less training cost and brings possibilities for online learning after the model is deployed.

In a word, to address this problem we need to find a control policy that could (a) maximize the coverage index and fairness index in the testing phase; (b) execute under partial observation and in a decentralized manner; (c) realize fully decentralized and distributed learning.

4 PRELIMINARIES

We consider the multi-UAV task as a sequence of parallel decision-making tasks. In the standard reinforcement learning(RL) framework, it can be formulated as *Markov game*, which is an extension of *Markov decision process* to multi-agent domains. Consider a Markov game of N agent, at each timeslot t , each agent i could obtain a local observation o_t^i , and takes an action a_t^i based on a policy $\pi^i : o^i \rightarrow a^i$. In each environmental timeslot, every agent executes its action simultaneously and obtains a corresponding reward r_t^i and the local observation o_{t+1}^i . The goal of each agent i in Markov game is to learn a policy that maximizes the discounted sum of future rewards:

$$J = \mathbb{E}_{o_t, a_t} [R_t] = \mathbb{E}_{o_t, a_t} [\sum_{t'=t}^T \gamma^{t'-t} r_{t'}] \quad (3)$$

where T is the final timeslot, and γ is the discount factor.

Q-Learning [37] is a classical method in RL. The key idea is to learn a Q-function that estimates the expected return

of the action taken by the current policy under the current state, i.e., $Q : o \times a \rightarrow R_t$. Now that a value function is obtained, a policy π can be designed by simply choosing the action with the highest estimated value, that is:

$$\pi(o) = \arg \max_a Q(o, a) \quad (4)$$

To approximate the expected return R_t , the Q-value function is recursively defined as $Q(o, a) = \mathbb{E}[r_t + \gamma \cdot \mathbb{E}[Q(o_{t+1}, a_{t+1})]]$, and is updated by minimizing the mean squared temporal difference error:

$$loss_Q = (r_t + \gamma \cdot \max_{a_{t+1}} Q(o_{t+1}, a_{t+1}) - Q(o_t, a_t))^2 \quad (5)$$

Deep Q Network(DQN) [38] is the first work that manages to use the deep neural network as the function approximator of the Q-function, which proposes two techniques to address the difficulty of convergence. Firstly, it introduces *target network* to calculate the learning target $y_t = r_t + \gamma \cdot \max_{a_{t+1}} Q'(o_{t+1}, a_{t+1})$, where Q' is the target network updated by copying the latest parameters of Q in every several iterations. Secondly, it adopts *experience replay* mechanism, which maintains a large sliding window to store the transition tuple (o_t, a_t, r_t, o_{t+1}) at each timeslot t . These two techniques have been proved to be effective to avoid divergence and stabilize training.

Due to the partial observable constraint, the observation o is a small subset of the global s . The intuition is that there may be some critical information that has been seen in the previous observation, but not in the current observation. Therefore, Deep Recurrent Q Network (DRQN) [39] introduced *recurrent neural network*(RNN) in the network structure to record the historical observations and alleviate the limitations of partial observation. Several later works [40] [41] also showed that RNN could effectively accelerate convergence and improve performance in the partial observable environment.

To reduce the training instability caused by training multiple policies simultaneously, MADDPG [17] introduced the centralized training and decentralized execution(CTDE) framework, which trains a full-observable value function (also known as the critic network) and a partial-observable decentralized policy for each agent. Specifically, the gradient of MADDPG's policy is:

$$\nabla J^i = \frac{1}{S} \sum_s \nabla (\pi(o_t^i) \cdot Q(o_t, a_t))|_{a_t^i = \pi^i(o_t^i)} \quad (6)$$

where S is the size of mini-batch; and o_t and a_t denote the joint observation and the joint action respectively. MADDPG concatenates all agents' observation and action into a long vector and takes it as the input of the Q-network, which is an inefficient method to aggregate information from all agents and leads to poor scalability [42].

The recent progress of deep learning has highlighted the performance of the *self-attention* mechanism [43], which provides an efficient method for aggregating the information in the multi-agent system. MAAC [42] is a popular MADRL algorithm based on CTDE paradigm that adopts self-attention mechanism. Firstly, it encodes the raw input of each agent; then it uses a bi-linear mapping called *query-key* system to calculate the similarity between every two embeddings (e_i, e_j) ; after that, it passes the similarities

into a softmax function for normalization. The normalized similarity between (i, j) is called *attention weight* α_{ij} , and can be represented as:

$$\alpha_{ij} = \frac{\exp(k_j^T \cdot q_i)}{\sum_{k=1}^N \exp(k_k^T \cdot q_i)} = \frac{\exp((W_k e_j)^T \cdot W_q e_i)}{\sum_{k=1}^N \exp((W_k e_k)^T \cdot W_q e_i)} \quad (7)$$

where W_q is a matrix that transforms the embedding e into a *query vector* q , and W_k transforms e into a *key vector* k . Then, use W_v to transform e into a *value vector*, and aggregate the value vector from all other agents according to the attention weights:

$$x_i = \sum_{j=1}^N \alpha_{ij} \cdot v_j = \sum_{j=1}^N \alpha_{ij} \cdot W_v e_j \quad (8)$$

Although MAAC solves the problem of poor scalability with the self-attention structure in the centralized critic, there is still room for improvement. On one hand, it requires information aggregation from **all** agents in the training phase, which is inefficient and computationally expensive. On the other hand, the MAAC policy only utilizes the local observation in the testing phase, hence its performance may be severely limited by the partial observation.

5 PROPOSED FULLY-DECENTRALIZED DRL SOLUTIONS FOR MULTI-UAV NAVIGATION

In this section, we present a fully decentralized DRL-based multi-UAV control solution for partial observable communication coverage. In the solution, all UAVs share the same policy for path planning and control in a decentralized manner with neighboring communication. Different from previous works that adopt the CTDE framework [17] [42] [44] and therefore require full observation during the training phase, our approach meets the partial observable constraints in both the training stage and the testing stage.

5.1 Observation Space and Action Space

For each UAV i at timeslot t , it could obtain a local observation o_t^i from the environment. As we designed in Sec. 3.1, the UAV i could observe the PoIs and other UAVs in a $(2R_{obs} + 1) \times (2R_{obs} + 1)$ square area centered at itself. The observation consists of three elements: PoI map $\mathbf{M}_{PoI}$, UAV map $\mathbf{M}_{UAV}$, and binary encoding of location $\mathbf{e}_{pos}$. PoI map and UAV map are vectors of length $(2R_{obs} + 1)^2$, which records the number of PoI and UAV at the corresponding position in the observable area. Binary encoding of location $\mathbf{e}_{pos}$ is a vector of length $2d$ that consist of the binary code of the UAV's current index on the grid map (x_t^i, y_t^i) , where d is a hyper-parameter that meets the condition that 2^d is greater than the map size L . Therefore the observation is a vector of length $2 \times (2R_{obs} + 1)^2 + 2d$, and could be represented as:

$$o_t^i = (\mathbf{M}_{PoI} | \mathbf{M}_{UAV} | \mathbf{e}_{pos}) \quad (9)$$

Where the operator $(\cdot | \cdot)$ denotes the concatenation operation.

The UAV's action space consists of 5 actions, including "upwards", "downwards", "leftwards", "rightwards", and "stay".

5.2 The Heuristic Reward Function

As is mentioned in Sec.3.2, the objective of the multi-UAV system is to maximize the global metrics including the final coverage index and final fairness index. A very intuitive and widely adopted method is to use these metrics to calculate the reward [15]. However, in our problem settings, the global metrics are not available in the training phase and we need to design a reward function that only uses the input features that can be obtained by the decentralized policy. We expect the reward function could encourage UAVs to achieve a better coverage index and fairness index. As the objective of DRL is to maximize the expected return R_t^i , ideally, we expect to design a reward function that meets:

$$\mathbb{E}[\sum_{i=1}^N R_t^i] \propto \mathbb{E}[f_t], \mathbb{E}[c_t] \quad (10)$$

As the role of the reward function is very similar to the role of *heuristic function* in A* algorithm, we name it as the *heuristic reward function*. The reward function consists of two parts. One is the *individual item*, defined as the number of PoIs that are exclusively covered by the agent. The second part is the *teamwork item*, defined as the number of PoIs covered by other UAVs in the one-hop adjacency graph of the UAV divided by the number of other drones. A hyper-parameter a is used to perform a weighted summation between two items. The reward function could be represented as:

$$r_i = a \times \#PoI(i, exclusive) + \frac{\#PoI(oneHop)}{\#oneHop} \quad (11)$$

where i is the UAV's index, *oneHop* is the set of other UAVs that are directly connected to the UAV i in the adjacency graph, the operator $\#$ denotes "the number of"; $\#PoI(x, exclusive)$ is a function that returns the number of PoI covered by x , when *exclusive* = *True*, it will return the number of PoIs that are exclusively covered by x . When $\#PoI$ returns zero, we additionally add a -1 penalty to the UAV so that the reward could be negative.

5.3 Deep Recurrent Graph Network

In this section, we present the Deep Recurrent Graph Network (DRGN) architecture, as shown in Fig. 2.

We use the *parameter-sharing* technique, i.e., all agents are controlled by the same neural network, which makes the model easy to converge, reduces training costs, and improves sample efficiency. Following [38], to decrease the computational complexity, for each agent, our network only takes the observation of the subgraph as input and outputs the Q-values for all possible actions $\mathbf{A}$, i.e., $Q : \mathcal{O} \rightarrow \mathbb{R}^A$.

Firstly, DRGN uses an encoder to encode the raw input:

$$e_i = ENC(o_i) \quad (12)$$

where o_i and e_i are the local observation and its embedding for agent i , and ENC is the encoder function shared by all agents. We use a fully connected layer as the encoder.

Then, we formulate the UAVs and their embeddings as a graph. Specifically, we regard each UAV i as a node and e_i as the node's embedding, and the topology of FANET is used to connect the nodes. In a graph convolutional layer, node embeddings can pass through the edge, so each node

Fig. 2. Network architecture of the proposed DRGN model. It consists of four modules: encoder, GAT layers, GRU, and MLP.

can obtain information from itself and neighboring nodes. For convenience, we define the set of a certain node i and its one-hop neighboring nodes as *subgraph* G_i , which could be represented as:

$$G_i = \{j | \forall j \text{ s.t. } A(i, j) = 1\} \quad (13)$$

where A is the adjacency matrix of FANET. The observations and embeddings of UAVs in G_i could be represented as $\mathcal{O}_i = \{o_j | \forall j \in G_i\}$ and $\mathcal{E}_i = \{e_j | \forall j \in G_i\}$, respectively.

In a subgraph G_i , to node i , the importance of each neighboring node may be different. For example, when the UAV i does not find any PoI in its observation area, the neighboring UAV with the biggest number of PoIs in its observation area should be of the greatest importance, while other nodes need less attention. To this end, we use *graph attention* (GAT) [45] Layer to process the graph data, which utilizes self-attention mechanism to decide the importance of each node in subgraph i by Equation 14, and aggregate the neighboring information for node i by Equation 15.

$$\begin{aligned} \alpha_{ij} &= \frac{\exp(a_{ij})}{\sum_{k \in G_i} \exp(a_{ik})} = \frac{\exp(k_j^T \cdot q_i)}{\sum_{k \in G_i} \exp(k_k^T \cdot q_i)} \\ &= \frac{\exp((W_K e_j)^T \cdot W_Q e_i)}{\sum_{k \in G_i} \exp((W_K e_k)^T \cdot W_Q e_i)} \end{aligned} \quad (14)$$

$$g_i = GAT(\mathcal{E}_i) = \sum_{j \in G_i} \alpha_{ij} \cdot v_j = \sum_{j \in G_i} \alpha_{ij} \cdot W_V e_j \quad (15)$$

Where α_{ij} is the attention weight that determines the importance of node j to node i , and g_i is the output embedding for node i after information aggregation. To improve the GAT's expressive capability, we implement the self-attention mechanism with M heads, i.e., to calculate Equation 15 M times in parallel, and concatenate the outputs:

$$g_i = \text{Concat}_{m=1}^M (g_i^m) \quad (16)$$

We stack two GAT layers to provide a two-hop perception region. As this structure is used for communication between UAVs, we named it as *GAT-based FANET* (GAT-FANET). To illustrate, the node i in the second GAT layer could obtain its neighbor nodes' embedding from the first GAT layer, which contains the information aggregated from the node that is two-hop to i . Note that the node i only communicates with its one-hop neighbors. This process is similar to the long-distance communication task in real life, where two nodes that are far apart could communicate through a relay station.

After GAT-FANET, each node i gets an embedding g'_i that aggregates information in the two-hop subgraph. Inspired by DRQN [39], we adopt a recurrent neural network (RNN) to record the historical graph embedding. Specifically, we choose *gated recurrent unit* (GRU) [46] as the recurrent unit:

$$h_t^i = GRU(g_t^i | h_{t-1}^i)$$

where h_{t-1}^i is the hidden state of UAV i 's GRU in the last timeslot. Finally, a linear transform is performed to the output of GRU to calculate the Q-values of each action.

5.4 Decentralized Training Process

The process of the UAVs interacting with the environment can be summarized as follows: At each timeslot, every UAV i obtains an observation o_t^i and calculates the Q-values for all actions. We use the ϵ -greedy strategy to choose the action a_t^i based on the Q-values. After the actions executed, each agent will obtain a reward r_t^i and a new observation o_{t+1}^i .

Following DQN [38], we use a target network for calculating the learning target of the model. The target network is a copy of the DRGN model. Its parameters are updated by directly copying the parameters of the trained model in every few iterations.

Specifically, we update the model by minimizing the loss function below:

$$\text{loss}_Q = \frac{1}{SN} \sum_{s=1}^S \sum_{i=1}^N (Q(\mathcal{O}_t, \mathcal{H}_t, a_t^i) - y_i)^2 \quad (17)$$

where $\mathcal{O}_t = \{o_t^j | \forall j \in G_t^i\}$, $\mathcal{H}_t = \{h_t^j | \forall j \in G_t^i\}$ and G_t^i is the subgraph i ; S is the size of mini-batch and N is the number of UAVs in the sampled experience. The learning target value y_i is calculated by $y_i = r_t^i + \gamma \max_{a_{t+1}^i} Q'(\mathcal{O}_{t+1}, \mathcal{H}_{t+1}, a_{t+1}^i)$, where Q' denotes the target network.

Theoretically, a tuple of $(\mathcal{O}_t^i, \mathcal{H}_t^i, a_t^i, r_t^i, \mathcal{O}_{t+1}^i, \mathcal{H}_{t+1}^i)$ is a fundamental unit for training DRGN model and we call it as an *experience*. According to the actual situation, we propose two strategies for replaying the experience. The first is designed for distributed training, it saves the experience of each agent individually and resamples it randomly, which is similar to other fully-decentralized algorithms. The second is designed for situations in which centralized training is feasible. It integrates the experience of all agents in a timeslot $(\mathbf{o}_t, \mathbf{h}_t, \mathbf{A}_t, \mathbf{a}_t, \mathbf{r}_t, \mathbf{o}_{t+1}, \mathbf{h}_{t+1}, \mathbf{A}_{t+1})$ and resamples it as a whole. In this way, it optimizes the efficiency of storing

Fig. 3. The flowchart of training phase and execution phase of our fully decentralized approach.

experiences, and since the sampled experiences belong to the same timeslot, the stability of the training process is improved.

Pseudocode for the training process of DRGN is presented in Figure 4.

Fig. 4. DRGN for the UAV-MBSs problem(centralized training version)

```

1: Initialize a same DRGN model  $Q$  of with parameters  $\theta$  and target network  $Q'$  with parameters  $\theta'$ .
2: Set global time step  $T \leftarrow 0$ 
3: for episode = 1  $\rightarrow$  max-episodes do
4:   Randomly reset the position of PoIs and UAVs.
5:   for local time step  $t = 1 \rightarrow$  episode-length do
6:      $T \leftarrow T + 1$ 
7:     for agent  $i = 1$  to  $N$  do
8:       Obtain the observations  $\mathcal{O}_t^i$  of the subgraph  $G_i$ 
9:       Select the action based on  $a_t^i = \arg \max_a Q(\mathcal{O}_t^i)$ 
10:      Execute the action  $a_t^i$  and obtain a reward  $r_t^i$  and observations  $\mathcal{O}_{t+1}^i$ 
11:      Obtain an experience  $(\mathcal{O}_t^i, \mathcal{H}_t^i, a_t^i, r_t^i, \mathcal{O}_{t+1}^i, \mathcal{H}_{t+1}^i)$ 
12:    end for
13:    Integrate and store the experience of all agents as:
        
$$(\mathbf{o}_t, \mathbf{h}_t, A_t, \mathbf{a}_t, \mathbf{r}_t, \mathbf{o}_{t+1}, \mathbf{h}_{t+1}, A_{t+1})$$

14:    if  $T \bmod \text{training-interval} = 0$  then
15:      Randomly sample  $S$  integrated experiences
16:      for Each integrated experience do
17:        Split it into  $N$  experiences using  $A_t$  and  $A_{t+1}$ :
            
$$(\mathcal{O}_t^i, \mathcal{H}_t^i, a_t^i, r_t^i, \mathcal{O}_{t+1}^i, \mathcal{H}_{t+1}^i)$$

18:      end for
19:      Use the  $S \times N$  experiences to update the DRGN model based on Equation 17.
20:    end if
21:    if  $T \bmod \text{target-update-interval} = 0$  then
22:      Update target network  $Q'$  by:  $\theta' \leftarrow \theta$ 
23:    end if
24:  end for
25: end for

```

6 EXPERIMENTS

6.1 Experimental Settings

We implemented our model using Pytorch 1.4.0, on a Ubuntu 18.04 server with 2 NVIDIA Titan RTX GPUs and 1 CPU of Intel Core i9-9900X @3.50GHz. The simulation environment is implemented in Python and accelerate by Numba's JIT technique. The target region is set as a 34×34 discrete grid map, with 100 PoIs that follow the Gaussian distribution. In the training phase, we deployed $N = 20$ UAVs with a shared DRL model in the simulation environment, and in the testing phase, the number of UAVs varies within $\{5, 10, 20, 40, 60, 80\}$. Without loss of generality, the coverage range R_{Cov} is 1 and the observe range R_{Obs} is 3. The maximum communication distance D_{Com} for FANET is 5. The length of each episode is 100, and when an episode ends, the positions of PoIs and UAVs will be reassigned.

We set the learning rate to $1e-4$ and use Adam [47] as the optimizer. The experience replay buffer size is $5e4$, and the batch size is 128. The neuron number of all hidden layers is 128, and all attention layers have 4 heads. The model is updated for 4 times in every 100 environmental timeslots. The target network is initialized with the same parameters as the model and is updated by copying the parameters of the model in every 500 environmental steps. The discounting factor γ is 0.99. The exploration strategy is $\epsilon - greedy$, in which the value of ϵ is initially 0.9 and exponentially decays to 0.3 at around 15,000 episode.

For each case, we train our models with 60,000 episodes. In every 100 episodes, an evaluation of the model is executed by running the frozen model with $\epsilon = 0.3$ for 100 episodes and calculating the averaged value of metrics.

We used the following global metrics for evaluation:

- 1) Final coverage index (c_T): This is the coverage index defined in Equation 1 at the end of the testing period, which measures the model's ability to explore and cover as many PoI as possible.
- 2) Final Fairness Index (f_T): This is the fairness index defined in Equation 2 at the last timeslot, which measures the fairness of the communication coverage provided by the UAVs.

6.2 State-of-the-art Approach and Baselines

We compared our approach, DRGN, with state-of-the-art DRL algorithms, including DQN, CommNet, MAAC, and DGN. DQN is a simple but strong decentralized approach for large-scale multi-agent tasks. MAAC is a recent algorithm that improves the scalability of MADDPG with the self-attention mechanism. Since our most relevant work [15] adopts MADDPG to control the multi-UAV navigation, we choose MAAC as the major object to be compared with. CommNet is a centralized approach that performs communications among all UAVs during training and execution. Thus we compare with it to show the superiority of our GAT-FANET communication structure. DGN is a recently proposed fully decentralized MADRL algorithm that also uses the GAT structure, we replace its GAT-layers with our designed GAT-FANET and compare it with DRGN to examine the necessity of GRU. The summary of the characteristics of DRL algorithms are given in Table 2

TABLE 2

Comparison of the characteristics of each algorithm. *DT* means decentralized training, and *DE* means decentralized execution, *Ad-hoc* denotes whether communicate during execution.

Algorithm	DT?	DE?	Ad-hoc?	Model-Free?
DQN	✓	✓	×	✓
MAAC	×	✓	×	✓
CommNet	×	×	✓	✓
DGN	✓	✓	✓	✓
DRGN(ours)	✓	✓	✓	✓
Random	/	✓	×	✓
MF-Greedy	/	✓	×	✓
MB-Greedy	/	✓	×	×
MB-GA	/	×	×	×

Meanwhile, we compared our approach with four heuristic baselines as:

- 1) **Random:** At each timeslot t , all UAVs randomly select a action.
- 2) **MF-Greedy:** At each timeslot t , each UAV obtains a local observation map and divides it into five sub-maps as: "Up", "Down", "Left", "Right", and "Center", then navigate in the direction to the sub-map that possess most PoIs. This policy only need the raw observation to decide the action, thus it is named as *Model-Free Greedy*(MF-Greedy).
- 3) **MB-Greedy:** At each timeslot t , this policy assumes that other UAVs choose "stay there", then tries all actions in a simulated environment to find the action a_t^i that could maximize the reward r_t^i . As a simulation environment that could estimate the reward of each action is required, this policy is named as *Model-Based Greedy*(MB-Greedy).
- 4) **MB-GA:** In this policy, the objective is to find a joint action $\mathbf{a}_t$ that maximizes the joint reward $\sum_i r_t^i$. As the joint action space is exponential to the number of UAVs, we adopted genetic algorithm(GA) to fit the near-optimal joint action at each timestep t . Since this policy also needs an environment model, we name it as *Model-Based GA*(MB-GA).

The characteristics of heuristics baselines are also summarized in Table 2.

6.3 Neural Network Convergence and Reward Function Effectiveness

As the multi-head GAT layer is the key element of DRGN and DGN, its convergence is crucial to ensure the performance of the model. Since GAT needs to calculate the attention weights of all nodes in a dynamic subgraph, we use GAT's attention weight to the node itself as an indicator to judge whether it has converged. As shown in Fig. 5, each head of GAT converges to a certain value, and the difference between the value of different heads proves the necessity of using multiple heads.

We then show the convergence of the tested DRL methods by showing the trend of evaluation reward over the training phase, which is shown in Fig. 6a. It also shows that DRGN converges faster and better than other algorithms.

Fig. 5. Attention weights on self after 240,000 iterations (60,000 episodes) of the model.

Regardless of the convergence of the model, another question is that since we use the heuristic reward function that only evaluates local performance to train the DRL models, the performance of the model in terms of global metrics is not guaranteed. As is illustrated in Section 5.2, Equation 10 should be fulfilled. As can be seen in Fig. 6, the final coverage index and final fairness index are improved together with the increase of the episodic reward, thus the effectiveness of the designed reward function is empirically demonstrated.

To be more intuitive, we run a converged DRGN model in the simulated environment for 500 timeslots, and show the trend of coverage index and fairness index during testing, as shown in Fig. 7. We observe that the initial fairness index value is around 0.46, then quickly reaches 0.9 at nearly 50 timeslots, thereafter converges to around 0.91 and never falls. Coverage index increases quickly in the first 100 timeslots, and keep rising slowly. These curves indicate that we can learn a good strategy for multi-UAV navigation with our designed reward function.

6.4 Finding Appropriate Structure for GAT-FANET

Next, we present the experimental results trying to find an appropriate network structure for GAT-FANET. We tried three structures as the communication structure and test them in DGN and DRGN: *one GAT layer*(one-hop), *two stacked GAT layers*(two-hop), and *two stacked GAT layers with skip-connection*(two-hop+SC). Note that vanilla DGN [35] uses two-hop+SC for inter-agent communication. As the reward curves are shown in Fig. 8, DGN is sensitive to the structure of GATs, and two-hop could achieve the best performance. On the contrary, DRGN is more robust to the GAT-structure as two-hop and two-hop+SC have close performance. It is possibly due to the fact that the recurrent structure in DRGN could store the history graph information. To keep the consistency of the GAT-FANET structure of DGN and DRGN, we use *two stacked GAT layers* for both of DGN and DRGN, as is shown in Fig. 2.

Fig. 6. Evaluation results of the model during a training phase of 60,000 episodes.

Fig. 7. Coverage index and fairness index over one testing episode.

6.5 Comparing with DRL and heuristic baselines

In this section, we evaluate the performance of our proposed approach DRGN and compare it with state-of-the-art DRL algorithms and heuristic baselines described in Sec.6.2.

We first compare DRGN with the other DRL methods. To make a comparison, for each algorithm, we choose the best-learned model and test it in environments with 20 UAVs for 100 episodes. From Figure 9, we can make the following observation:

Firstly, in terms of all metrics, MAAC has demonstrated its performance beyond DQN. As both of them are decentralized policies that only require local observation, this may be because MAAC's policy is trained by the centralized critic with a global observation. In addition, DGN and DRGN, which are decentralized policies with GAT-FANET communication, showed better performance than the decentralized policies of DQN and MAAC, as well as the centralized policy of CommNet. Furthermore, the comparison between DRGN and DGN denotes that the RNN structure can improve the performance in the partial observable task.

Secondly, in Fig. 9c, DRGN and DGN have a smaller variance in fairness index than the decentralized policies of DQN and MAAC. From our perspectives, this may be because when there is no PoI in local observation, the decentralized policy can only move randomly, while policies with UAV-FANET can obtain information from the neighbor and navigate to the place with the most uncovered PoIs.

(a) Learning curves of DGNs.

(b) Learning curves of DRGNs.

Fig. 8. Episodic reward curves of DGN/DRGN with different communication structure.

Then we compared DRGN with the heuristic baselines. All heuristic policies are tested in environments with 20 UAVs for 100 episodes. As shown in Fig. 10, although our proposed DRGN is a model-free policy, it not only outperforms other hand-crafted model-free policies (Random and MF-Greedy) but also shows superior performance to the model based-policies (MB-Greedy and MB-GA). Also, it can be seen that the result of DRGN has a smaller variance than other heuristic policies, which means that DRGN is more robust to different environments.

Fig. 9. The Evaluation results by testing the best-learned model of each DRL algorithm in the environment with 20 UAVs for 100 episodes.

Fig. 10. The Evaluation results by testing the heuristics policies in the environment with 20 UAVs for 100 episodes.

6.6 Scalability and Transfer-ability

Although previous work could effectively control the multi-UAV navigation, the scale of the UAV swarm is often only about 5-10. Such a small scale of UAVs may not be sufficient to meet the needs of real missions, and the learned model needs to be tested with a larger group of UAVs. Moreover, in real-world tasks, due to emergencies such as energy shortage or being attacked, the number of UAVs may change dynamically. Therefore, it is necessary to compare the performance of the trained model under different UAV scales. To this end, we tested the performance of DRL methods and heuristic baselines with the number of UAVs varies within [5, 10, 20, 40, 60, 80].

We first compared the scalability and transferability of the DRL methods. During the training phase of the DRL models, there are consistently 20 UAVs in the environment; while in the testing phase, the scale of UAVs is modified to different numbers. For each case, we tested 100 episodes and calculated the mean and standard deviation, as shown in Fig. 11. By experiments, we have the following observations:

- 1) As the number of UAVs grows, the performance of DRGN increases, validating the scalability of DRGN. Besides, regardless of the number of UAVs, DRGN performs consistently better than other DRL methods, verifying a good transferability of DRGN.
- 2) As shown in Fig. 11a, when the number of UAVs is greater than 20, the reward for all models begins to drop. According to Equation 11, this is partially due to the difficulty of covering a PoI exclusively

when the number of UAVs is large, making the individual item drop; meanwhile, the scale of the Ad-hoc network becomes larger and the average number of PoIs covered per UAV decreases, making the teamwork item decrease.

We then compare DRGN with the heuristic methods. From Fig. 12, we can observe that the transfer-deployed DRGN model shows better performance than the model-based baselines, demonstrating its good transfer-ability. Considering the time efficiency, we record the inference time for each policy to infer a joint action a_t . As shown in Fig. 12c, DRGN has an execution speed close to other model-free methods, which is about 1000x of MB-GA. To be intuitive, it takes about 325 hours to use MB-GA to control 60 UAVs for 100 episodes. By contrast, DRGN only takes about 20 minutes.

6.7 Robustness and Interpretability of GAT-FANET

After verifying the performance of DRGN, we now dive deeper into the performance and the principle of the GAT-FANET, which is a key element in DRGN.

We first examine the robustness of GAT-FANET. In the training phase, we assume that the communication is completely reliable. However, in the real world, communications within the Ad-hoc network can be interrupted for various reasons. Thus, we assume that the communication between adjacent nodes in the FANET is interrupted with probability p , and tested the performance of DGN and DRGN under different p . It can be seen from Fig. 13 that DGN and

Fig. 11. The evaluation results of testing each best-learned DRL model in the environment with different number of UAV for 100 episodes.

Fig. 12. The evaluation results of testing each heuristic policy in the environment with different number of UAV for 100 episodes.

DRGN have good robustness on the communication loss of the FANET. Note that even if the communication is completely lost, i.e., our proposed GAT-FANET based DRGN and DGN are only granted with local observation, they still show competitive performance to model-based baselines and outperform the decentralized DRL policies.

We then consider the interpretability of the GAT network. We selected an interesting moment when testing the DRGN model, as shown in Fig. 14a, which is a decisive step for UAV0 (marked as red circle). Due to the partial observation limitations, UAV0 needs to pay attention to the information from UAV7 that there is a large amount of PIs below to make a decision to move downwards. To figure out, we visualize the attention weights of UAV0's second GAT layer at this moment, as shown in Fig. 14b. Apart from holding 0.96 attention to itself in head2, UAV0 also has 0.6 attention to UAV7, while other neighboring UAVs only have 0.29 attention at most. This denotes that GAT-FANET has learned the ability to automatically identify valuable neighbors.

7 CONCLUSION

In this paper, UAV-MBSs is redefined as a partially observable problem, in which global information is not available during the training phase. To solve this problem, we propose a novel DRL method named DRGN, which uses a GAT-based Ad-hoc network and a recurrent unit GRU to alleviate the limitations of partial observation. DRGN does

not require global information during the training phase, which enables distributed training and greatly reduces the training cost. Through extensive experiments, we designed a suitable network structure for GAT-FANET and demonstrated the performance superiority of DRGN compared to other DRL methods and heuristic methods. Furthermore, the scalability, transferability, robustness, and interpretability of DRGN have been demonstrated. As for future work, we would like to extend DRGN to the actor-critic style to achieve continuous action space.

ACKNOWLEDGMENTS

This work is supported by the National key R & D program of China sub project "Emergent behavior recognition, training and interpretation techniques" under grant No. 2018AAA010230.

REFERENCES

- [1] A. A. Süzen, B. Duman, and B. Şen, "Benchmark analysis of jetson tx2, jetson nano and raspberry pi using deep-cnn," in *2020 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*. IEEE, 2020, pp. 1–5.
- [2] L. Meier, D. Honegger, and M. Pollefeys, "Px4: A node-based multi-threaded open source robotics framework for deeply embedded platforms," in *2015 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2015, pp. 6235–6240.
- [3] J. Wu and I. Stojmenovic, "Ad hoc networks," *Computer*, vol. 37, no. 2, pp. 29–31, 2004.

Fig. 13. The evaluation results of testing DGN and DRGN in the environment with 20 UAVs for 100 episodes, with the communication drop probability ranging from 0 to 1.

(a) A screenshot of a critical timeslot for UAV0 (the red one). The one-hop neighbors of UAV0, including UAV{2,5,7,15,16}, are marked as green circles. The green blocks represent the randomly distributed Pols and blue circles are other UAVs.

(b) The attention weights of UAV0's second GAT layer at the specific timeslot shown in (a).

Fig. 14. An example for illustrate the intrepretability of DRGN and its GAT-based Ad-hoc network.

- [4] M. Bhaskaranand and J. D. Gibson, "Low-complexity video encoding for uav reconnaissance and surveillance," in *2011-MILCOM 2011 Military Communications Conference*. IEEE, 2011, pp. 1633–1638.
- [5] H. Menouar, I. Guvenc, K. Akkaya, A. S. Uluagac, A. Kadri, and A. Tuncer, "Uav-enabled intelligent transportation systems for the smart city: Applications and challenges," *IEEE Communications Magazine*, vol. 55, no. 3, pp. 22–28, 2017.
- [6] Y. Zhang, X. Yuan, W. Li, and S. Chen, "Automatic power line inspection using uav images," *Remote Sensing*, vol. 9, no. 8, p. 824, 2017.
- [7] W. Li, B. Yang, G. Song, and X. Jiang, "Dynamic value iteration networks for the planning of rapidly changing uav swarms," *Frontiers of Information Technology & Electronic Engineering*, pp. 1–10, 2021.
- [8] M. Moradi, K. Sundaresan, E. Chai, S. Rangarajan, and Z. M. Mao, "Skycore: Moving core to the edge for untethered and reliable uav-based lte networks," in *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*, 2018, pp. 35–49.
- [9] L. Zhong, K. Garlich, S. Yamada, K. Takano, and Y. Ji, "Mission planning for uav-based opportunistic disaster recovery networks," in *2018 15th IEEE Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2018, pp. 1–6.
- [10] Z. Li, Y. Wang, M. Liu, R. Sun, Y. Chen, J. Yuan, and J. Li, "Energy efficient resource allocation for uav-assisted space-air-ground internet of remote things networks," *IEEE Access*, vol. 7, pp. 145 348–145 362, 2019.
- [11] A. Merwaday and I. Guvenc, "Uav assisted heterogeneous networks for public safety communications," in *2015 IEEE wireless communications and networking conference workshops (WCNCW)*. IEEE, 2015, pp. 329–334.
- [12] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Unmanned aerial vehicle with underlaid device-to-device communications: Performance and tradeoffs," *IEEE Transactions on Wireless Communications*, vol. 15, no. 6, pp. 3949–3963, 2016.
- [13] M. Alzenad, A. El-Keyi, and H. Yanikomeroglu, "3-d placement of an unmanned aerial vehicle base station for maximum coverage of users with different qos requirements," *IEEE Wireless Communications Letters*, vol. 7, no. 1, pp. 38–41, 2017.
- [14] C. H. Liu, Z. Chen, J. Tang, J. Xu, and C. Piao, "Energy-efficient uav control for effective and fair communication coverage: A deep reinforcement learning approach," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 9, pp. 2059–2070, 2018.
- [15] C. H. Liu, X. Ma, X. Gao, and J. Tang, "Distributed energy-efficient multi-uav navigation for long-term communication coverage by deep reinforcement learning," *IEEE Transactions on Mobile Computing*, vol. 19, no. 6, pp. 1274–1285, 2019.
- [16] T. Lillicrap, J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *CoRR*, 09 2015.
- [17] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Advances in neural information processing systems*, 2017, pp. 6379–6390.
- [18] L. Li, Z. Gan, Y. Cheng, and J. Liu, "Relation-aware graph attention network for visual question answering," in *Proceedings of the*

- IEEE/CVF International Conference on Computer Vision, 2019, pp. 10 313–10 322.
- [19] L. Ruan, J. Wang, J. Chen, Y. Xu, Y. Yang, H. Jiang, Y. Zhang, and Y. Xu, "Energy-efficient multi-uav coverage deployment in uav networks: A game-theoretic framework," *China Communications*, vol. 15, no. 10, pp. 194–209, 2018.
- [20] M. De Benedetti, F. D'Urso, G. Fortino, F. Messina, G. Pappalardo, and C. Santoro, "A fault-tolerant self-organizing flocking approach for uav aerial survey," *Journal of Network and Computer Applications*, vol. 96, pp. 14–30, 2017.
- [21] X. Zhang and L. Duan, "Fast deployment of uav networks for optimal wireless coverage," *IEEE Transactions on Mobile Computing*, vol. 18, no. 3, pp. 588–601, 2018.
- [22] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Mobile unmanned aerial vehicles (uavs) for energy-efficient internet of things communications," *IEEE Transactions on Wireless Communications*, vol. 16, no. 11, pp. 7574–7589, 2017.
- [23] I. Bekmezci, O. K. Sahingoz, and S. Temel, "Flying ad-hoc networks (fanets): A survey," *Ad Hoc Networks*, vol. 11, no. 3, pp. 1254–1270, 2013.
- [24] A. Joshi, S. Dhongdi, S. Kumar, and K. Anupama, "Simulation of multi-uav ad-hoc network for disaster monitoring applications," in *2020 International Conference on Information Networking (ICOIN)*. IEEE, 2020, pp. 690–695.
- [25] P. Li and H. Duan, "A potential game approach to multiple uav cooperative search and surveillance," *Aerospace Science and Technology*, vol. 68, pp. 403–415, 2017.
- [26] F. Ropero, P. Muñoz, and M. D. R-Moreno, "Terra: A path planning algorithm for cooperative ugv-uav exploration," *Engineering Applications of Artificial Intelligence*, vol. 78, pp. 260–272, 2019.
- [27] N. Mahdoui, V. Frémont, and E. Natalizio, "Communicating multi-uav system for cooperative slam-based exploration," *Journal of Intelligent & Robotic Systems*, vol. 98, no. 2, pp. 325–343, 2020.
- [28] S. Hayat, E. Yanmaz, T. X. Brown, and C. Bettstetter, "Multi-objective uav path planning for search and rescue," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 5569–5574.
- [29] S.-Y. Park, C. S. Shin, D. Jeong, and H. Lee, "Dronetx: Network reconstruction through connectivity probing and relay deployment by multiple uavs in ad hoc networks," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 11, pp. 11 192–11 207, 2018.
- [30] H. Shiri, J. Park, and M. Bennis, "Massive autonomous uav path planning: A neural network based mean-field game theoretic approach," in *2019 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2019, pp. 1–6.
- [31] L. Buşoniu, R. Babuška, and B. De Schutter, "Multi-agent reinforcement learning: An overview," *Innovations in multi-agent systems and applications-1*, pp. 183–221, 2010.
- [32] P. Rui, "Multi-uav formation maneuvering control based on q-learning fuzzy controller," in *2010 2nd International Conference on Advanced Computer Control*, vol. 4. IEEE, 2010, pp. 252–257.
- [33] S.-M. Hung and S. N. Givigi, "A q-learning approach to flocking with uavs in a stochastic environment," *IEEE transactions on cybernetics*, vol. 47, no. 1, pp. 186–197, 2016.
- [34] Y. Zhang, Z. Mou, F. Gao, J. Jiang, R. Ding, and Z. Han, "Uav-enabled secure communications by multi-agent deep reinforcement learning," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 10, pp. 11 599–11 611, 2020.
- [35] J. Jiang, C. Dun, T. Huang, and Z. Lu, "Graph convolutional reinforcement learning," *arXiv preprint arXiv:1810.09202*, 2018.
- [36] A. Dai, R. Li, Z. Zhao, and H. Zhang, "Graph convolutional multi-agent reinforcement learning for uav coverage control," in *2020 International Conference on Wireless Communications and Signal Processing (WCSP)*, 2020, pp. 1106–1111.
- [37] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3–4, pp. 279–292, 1992.
- [38] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.
- [39] M. Hausknecht and P. Stone, "Deep recurrent q-learning for partially observable mdps," in *2015 aaai fall symposium series*, 2015.
- [40] R. E. Wang, M. Everett, and J. P. How, "R-maddpg for partially observable environments and limited communication," *arXiv preprint arXiv:2002.06684*, 2020.
- [41] T. Rashid, M. Samvelyan, C. Schroeder, G. Farquhar, J. Foerster, and S. Whiteson, "Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2018, pp. 4295–4304.
- [42] S. Iqbal and F. Sha, "Actor-attention-critic for multi-agent reinforcement learning," in *ICML*, 2019.
- [43] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.
- [44] J. Ackermann, V. Gabler, T. Osa, and M. Sugiyama, "Reducing overestimation bias in multi-agent domains using double centralized critics," *arXiv preprint arXiv:1910.01465*, 2019.
- [45] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.
- [46] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.
- [47] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization."

Zhenhui Ye He received the B.S. degree from Zhejiang University, China, in 2020. He is currently pursuing a M.Sc. degree in aerospace information technology with the School of Aerospace and Astronautics at Zhejiang University. His research interests include practical reinforcement learning and deep learning in real-world applications.

Ke Wang He received the B.S. degree from Zhejiang University of Technology, China, in 2020. He is currently pursuing a Ph.D. degree in aerospace information technology at Zhejiang University. His research interests include multi-agent reinforcement learning and Software-Defined Networking.

Yining Chen Yining Chen was born in Hangzhou, China. He received the B.Sc. degree from Sichuan University, China in 2015. He is currently a Ph.D. student at Zhejiang University. His research interests include reinforcement learning and multi-robot system.

Xiaohong Jiang Xiaohong Jiang received her B.Sc. and M.Sc. degree in computer science from Nanjing University and the Ph.D. degree in Zhejiang University, China, in 2003. She is an associate Professor at the College of Computer Science and Technology, Zhejiang University. Her research focuses on distributed systems, cloud computing, and data service.

Guanghua Song Guanghua Song received the B.S. degree in Computer Science from Nanjing University of Science and Technology, China, in 1989, and the M.S. and Ph. D. degrees in Computer Science from Zhejiang University, China, in 1992 and 2003, respectively. He is currently a full Professor with School of Aeronautics and Astronautics, Zhejiang University, China. His research interests include swarm intelligence, UAV intelligence, and aerospace information technology.