Air-Ground Spatial Crowdsourcing with UAV Carriers by Geometric Graph Convolutional Multi-Agent Deep Reinforcement Learning

Yu Wang, Jingfei Wu, Xingyuan Hua, Chi Harold Liu, Guozheng Li, Jianxin Zhao, Ye Yuan, Guoren Wang
School of Computer Science and Technology, Beijing Institute of Technology, China
{yuwang0527, jingfeiwu, xingyuanhua, chiliu, guozheng.li, jianxin.zhao, yuan-ye, wanggr}@bit.edu.cn

2023 IEEE 39th International Conference on Data Engineering (ICDE) | 979-8-3503-2227-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICDE55515.2023.00140

***Abstract*—Spatial Crowdsourcing (SC) has been proved as an effective paradigm for data acquisition in urban environments. Apart from using human participants, with the rapid development of unmanned vehicles (UVs) technologies, unmanned aerial or ground vehicles (UAVs, UGVs) are equipped with various high-precision sensors, enabling them to become new types of data collectors. However, UGVs' operational range is constrained by the road network, and UAVs are limited by power supply, it is thus natural to use UGVs and UAVs together as a coalition, and more precisely, UGVs behave as the UAV carriers for range extensions to achieve complicated air-ground SC tasks. In this paper, we propose a novel communication-based multi-agent deep reinforcement learning method called "GARL", which consists of a multi-center attention-based graph convolutional network (GCN) to accurately extract UGV specific features from UGV stop network called "MC-GCN", and a novel GNN-based communication mechanism called "E-Comm" to make the cooperation among UGVs adaptive to constant changing of geometric shapes formed by UGVs. Extensive simulation results on two campuses of KAIST and UCLA campuses show that GARL consistently outperforms eight other baselines in terms of overall efficiency.**

***Index Terms*—Spatial crowdsourcing, multi-agent reinforcement learning, graph neural network**

I. Introduction

Spatial crowdsourcing (SC [1]–[4]) is an attractive paradigm that assigns a particular spatial-temporal task to a collective group of workers (normally human participants), which has many applications in ride hailing services [5], road condition monitoring [6], etc. To achieve higher flexibility and lower latency, instead of using human participants, unmanned vehicles (UVs) including unmanned aerial vehicles (UAVs) and unmanned ground vehicles (UGVs) have shown great potential to complete SC tasks [7]–[9], since they are equipped with high-speed data receivers like WiFi/5G to enable sensory data collection and sharing in urban environment. However, since UGVs cannot reach high-up areas and UAVs are usually associated with inadequate operational range due to limited power supply, it is challenging to simply utilize one type of UVs only. Therefore, it is promising to use UGVs as "carriers" and allow UAVs to land on top of them as a *coalition* to achieve data collection services, which helps overcome their individual constraints and thus improve overall task efficiency.

The coalition of UGVs and UAVs shows potentials in several SC tasks with UVs, such as disaster response [10] and

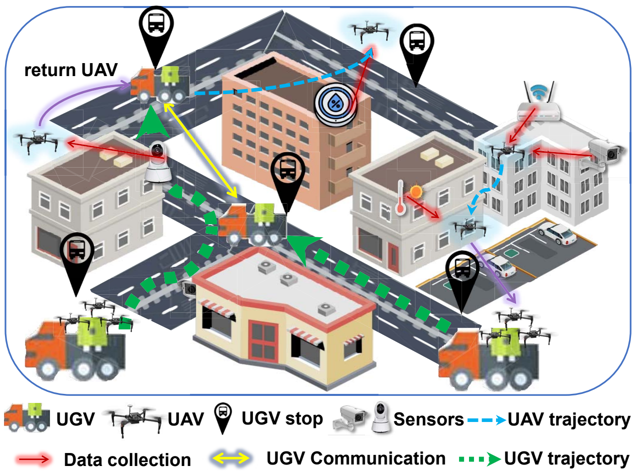


Fig. 1. Overall scenario for air-ground SC with UAV carriers.

daily surveillance [11]. In these tasks, the UAVs-UGV coalition can be deployed to support environmental awareness by collecting valuable data from scattered point-of-interests (PoIs) such as cameras and sensors for environmental inspection and monitoring. The UGVs are responsible for identifying the data-rich areas in complex environments and carrying UAVs to these areas. Then, UAVs are released to collect data from these PoIs. As an example scenario shown in Fig. I, several UGV stops are located along the roads, and UAVs are loaded. UGVs also provide battery charging services for UAVs and decide which stop to go next and when to release the UAVs to fly over the workzone to collect data from a variety of deployed sensors like CCTV cameras and gas alarms. To the best of our knowledge, we are one of the first to consider air-ground SC with UAV carriers.

In an air-ground SC task with UAV carriers, key challenges are, first, rapid and unpredictable changes of the sensory data to be collected may cause difficulties for UAVs-UGV coalitions to have an overall understanding on the entire environment conditions in real-time. Along the direction of spatial modeling, existing research on graph neural networks (GNN) related methods [12], [13] can capture the features of UGV stops as graph nodes, which is constructed using road connectivity. However, in practice, the entire workzone can be really large, even if not, it is also impossible for one single UGV to know the entire graph since it has partial observations of the environment, which may in turn be resulted in overlapping/missed visits to certain sensors. Therefore, it is natural to use communication-based multi-agent deep reinforcement learning (MADRL) as the start point of the design to control UGVs/UAVs, and many efforts have been paid. GAM [14] and IC3Net [15] improved the performance in cooperative scenarios by designing specific communication mechanisms.

Along with this direction, we raise the second challenge that existing communications mechanisms are not adaptive in environments where the data distribution is not uniform among sensors. In detail, according to equivariance theories [16], [17], existing communications mechanisms are based on GNN models, which are invariant to translation and rotation of geometric attributes of graph nodes (i.e., the UGVs in this paper). Thus, the communication-based MADRL models such as CommNet [18] cannot adapt to the constant changing of geometric shapes formed by UGVs. However, these geometric changes should be carefully considered when dealing with complex environments. For example, in an environment with uneven data distributions, the shape formed by UGVs at a time may keep similar after geometric changes, but UGVs should behave differently since the data density is changed. Hence, existing solutions lack an efficient communication mechanism for UAVs-UGV trajectory planning, which can efficiently exchange messages among UGVs, and be sensitive to geometric changes simultaneously.

To this end, in this paper, we propose a novel MADRL model called "GARL", which extracts UGV specific features from the stop network based on an attention module, and supports efficient geometry-aware message exchange among UGVs, using IPPO [19] as the start point of the design. Our contribution is three-fold:

- We propose a novel multi-center attention-based graph convolutional network called "MC-GCN" to extract UGV specific features of stop network from their own observations of the environment.
- We propose a geometry-aware communication mechanism called "E-Comm", which well adapts to constant changing of geometric shapes formed by UGVs including transformation and rotation.
- We perform extensive experiments on two campuses: KAIST (South Korea) and UCLA (USA). We find the most appropriate hyperparameters, and conduct ablation study, visualize the UGV/UAV trajectory, and show performance comparisons with eight baselines. Results confirm that "GARL" outperforms all others in term of overall efficiency.

The remainder of this paper is organized as follows. First, we review related activities in Section II. Then, we present the system model and give problem definition and formulation in Section III. Next, we present our proposed method GARL in Section IV. Experimental results are provided in Section V and we conclude the paper in Section VI. We list the important notations in Table I.

TABLE I
IMPORTANT NOTATIONS USED IN THE PAPER.

Notation	Explanation
t, T	Time index and task duration.
$\mathcal{U}, u, U$	UGV set, index and total number of UGVs.
$\mathcal{V}, v, V$	UAV set, index and total number of UAVs.
V'	The number of UAVs on each UGV.
$\triangle d_t^{v,p}$	Data collected by a UAV v from sensor p during timeslot $[t, t+1)$.
d_t^p	Remaining amount of data of sensor p at t.
$\lambda, \psi, \xi, \zeta, \beta$	Efficiency, data collection ratio, fairness, cooperation factor and energy consumption ratio.
$\boldsymbol{x}_t^b$	UGV stop vector of node b at t.
$\boldsymbol{x}_t^u$	UGV u's position at t.
$\boldsymbol{s_t}$	State at t.
$\boldsymbol{o}_t^u, \boldsymbol{o}_t^v$	Observation of UGV u and UAV v at t.
$\boldsymbol{a}_t^u, \boldsymbol{a}_t^v$	Action of UGV u and UAV v at t.

II. Related Work and Preliminaries

A. Spatial Crowdsourcing (SC)

With the development of high-speed mobile Internet and the increasing popularity of smart devices, people can easily get access to online services and engage in spatial-temporal cooperative tasks [20]. A new framework, namely spatial crowdsourcing [21] is proposed to support these real-world activities. In SC, many existing solutions are proposed. For example, Ni *et al.* in [22] considered task dependencies, where one task can be assigned only if its dependent tasks have been dispatched. Based on this constraint, they proposed greedy and game-theoretic approaches. Cheng *et al.* in [23] considered worker cooperations, and they proposed a Cross Online Matching algorithm (COM), making a platform to borrow available workers from other platforms for finishing requests from users. Besides, they also proposed two algorithms, including deterministic cross online matching and randomized cross online matching for COM.

In SC with UVs, Zhou *et al.* investigated joint optimization of route planning and task assignment problem for the sake of energy efficiency in UAV-aided SC systems [24]. Liu *et al.* developed a distributed DRL-based framework to achieve energy-efficient multi-UAV navigation, ensuring long-term communication coverage [25]. Wang *et al.* proposed FD-MAPPO (Cubic Map) [11] to enable humans to work collaboratively with UAVs to achieve data collection tasks based on a fully decentralized MADRL framework. In air-ground SC, the combination of multiple types of UVs helps expand the range of workzone and as a result improves the task completion ratio. IADRL [26] used a coalition of a UGV and a UAV to complete certain tasks by combining imitation learning with DRL. However, the tight binding between one single UGV and one UAV is not optimal. Wang *et al.* in [27] investigated Space-Air-Ground Integrated Networks (SAGINs) which dynamically reorganizes resources to meet the requirements of different services. The crowdsourcing methods are also considered in the SAGINs in order to complete complex and large-scale sensing tasks.

B. Graph Neural Network (GNN)

GNN methods operate on graph structured data, learning graph representation through node and edge features. Message passing scheme is the key part of these models, determining how a node aggregates information from their neighborhood in a graph. Graph convolutional networks (GCNs [12]) are the de facto method for graph processing and many related methods have been proposed to extract locally connected features from graphs [28], [29], and widely used in many applications, such as social networks [30], molecule structure modeling [31] and traffic prediction [32]. GCN is built by interleaving vertex-wise operations, implemented via a single fully-connected layer, with a communication step exploiting the Laplacian matrix of the graph. In practice, a single GCN layer provides a weighted combination of information across neighbors, representing a localized 1-hop exchange of information. A generic GCN layer can be described as:

$$\boldsymbol{X}^{(l+1)} = \sigma(\boldsymbol{L}\boldsymbol{X}^{(l)}\boldsymbol{W}), \tag{1a}$$

$$\boldsymbol{L} = \tilde{\boldsymbol{D}}^{-\frac{1}{2}}\tilde{\boldsymbol{A}}\tilde{\boldsymbol{D}}^{-\frac{1}{2}}, \tag{1b}$$

where $\tilde{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{I}_N$ is the adjacent matrix with self-connection; $\boldsymbol{L}$ is the Laplace matrix and $\tilde{\boldsymbol{D}}$ is a diagonal matrix where $\tilde{\boldsymbol{D}}_{ii} = \sum_j \tilde{\boldsymbol{A}}_{ij}$; $\boldsymbol{X}^{(l)}$ is the node feature matrix on l-th GCN layer and $\boldsymbol{W}$ is a learnable weight matrix.

Message passing neural network (MPNN) framework [33] concludes the message passing schemes of existing neural network models for graph structured data. An MPNN layer is operated in message passing phase and readout phase. For the former, each node in the graph receives and combines messages from its neighbor nodes through corresponding edges linked between them. Then, the features of all nodes are updated as the readout results according to their received messages.

C. Multi-Agent Deep Reinforcement Learning (MADRL)

In cooperative multi-agent tasks, a decentralized partially observable Markov decision process (Dec-POMDP) [34] is used to describe partially observable environment. A Dec-POMDP is formally defined as a tuple $\langle \mathcal{U}, \mathcal{S}, \mathcal{A}, Pr, R, \mathcal{O}, \rho, \rho_0, \gamma \rangle$, where $\mathcal{U}$, $\mathcal{S}$, $\mathcal{A}$ and $\mathcal{O}$ are the set of agents, states, actions and observations. The initial state $\boldsymbol{s}_0$ is drawn from distribution ρ_0. During timeslot $[t, t+1)$, each agent $u \in \mathcal{U}$ obtains its observation $\boldsymbol{o}_t^u \in \mathcal{O}$, and then chooses its action $\boldsymbol{a}_t^u$ based on local observation following its policy. After all agents taking the joint actions $\{\boldsymbol{a}_t^u\}_{u=1}^U$, the next state $\boldsymbol{s}_{t+1}$ is drawn from transition probability kernel $Pr(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, \{\boldsymbol{a}_t^u\}_{u=1}^U)$. Then, each agent u receives a individual reward $r_t^u = R(\boldsymbol{s}_t, \{\boldsymbol{a}_t^u\}_{u=1}^U, u)$ from the environment. The aim is to maximize the discounted accumulative reward $J(\theta) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t \sum_{u=1}^U r_t^u]$, where $\gamma \in [0, 1)$ is the discount factor.

Using deep methods [35] along with MARL techniques (MADRL) is becoming a powerful paradigm to study multi-agent problems. Centralized and decentralized learning are two general frameworks in MADRL algorithms. To mitigate the instability in decentralized methods, centralized training and decentralized execution (CTDE) framework is applied [36]–[38]. However, these models still lack of ability in message exchanges among agents, especially in partially observable and stochastic environment.

Both IPPO [19] and MAPPO [39] show competitive performance in cooperative tasks. IPPO learns independently in a fully decentralized manner while MAPPO estimates decentralized actor and centralized critic based on global state. Despite of the different frameworks, IPPO and MAPPO have similar objective functions in general, which are defined as:

$$L^{CLIP+VF+H}(\boldsymbol{\theta}^u) = \mathbb{E}_t\big[L_t^{CLIP}(\boldsymbol{\theta}^u) - c_1 L_t^{VF}(\boldsymbol{\theta}^u) + c_2 H(\pi_{\boldsymbol{\theta}^u})\big], \tag{2}$$

where $L_t^{CLIP}(\boldsymbol{\theta}^u)$ and $L_t^{VF}(\boldsymbol{\theta}^u)$ denote the clipped policy loss for actor and the value function loss for critic, respectively; $H(\pi_{\boldsymbol{\theta}^u})$ denotes the entropy loss of policy $\pi_{\boldsymbol{\theta}^u}$; c_1 and c_2 are constants that control the effects of the losses.

III. System Model and Problem Formulation

A. System Model

In our considered air-ground SC tasks with UAV carriers, let $\mathcal{U} \triangleq \{1, ..., u, ..., U\}$ and $\mathcal{V} \triangleq \{1, ..., v, ..., V\}$ denote UGVs and UAVs in the target workzone, respectively. Each UAV v is associated with limited initial fully-charged battery supply e_0^v. UAVs which are loaded on a UGV u are denoted as $\mathcal{V}^u \triangleq \{v | v \in \mathcal{V}, v \text{ is carried by } u\}$, where UGV u can replenish the battery of carried UAVs in $\mathcal{V}^u$ if needed. Let V' denote the number of UAVs carried by each UGV. In a task, UGVs and UAVs cooperate to collect data from sensors $\mathcal{P} \triangleq \{1, ..., p, ..., P\}$, which distribute around buildings in the target workzone. Initially, a sensor p contains $d_0^p, \forall p \in \mathcal{P}$ unit data to be collected by UAVs. Due to the limitation of the flight height and prohibited areas of UAVs by regional regulations or rules, for ease of exposition, we regard all the buildings as obstacles that UAVs cannot fly over. UGVs travel along the stop network in the target workzone, which can be abstracted into a graph called "stop graph" denoted as $\mathcal{G} = \{\mathcal{B}, \mathcal{E}\}$. Operations to construct a UGV stop graph are as follows: first, virtual UGV stop nodes $\mathcal{B} \triangleq \{1, ..., b, ..., B\}$ are set at regular intervals along the roads, and then the edges $\mathcal{E}$ in $\mathcal{G}$ are connected according to the connectivity of these virtual nodes in practice.

Without loss of generality, we consider a time-slotted system, dividing the task duration into T timeslots. In each timeslot $[t, t+1)$, each UGV u first decides whether to *release* UAVs in $\mathcal{V}^u$ to collect data in the surrounding area or not. If a UGV u decides to do so, it flies the UAVs and then wait in its position $\boldsymbol{x}_t^u = (x_t^u, y_t^u)$ for a specified period t^{rls}; otherwise, a UGV u chooses the next stop and moves over. During each timeslot, a UGV u moves a certain distance $\delta_t^u \leq \delta_{\max}^u$, and each released UAV v moves a certain distance $\delta_t^v = \sqrt{(x_{t+1}^v - x_t^v)^2 + (y_{t+1}^v - y_t^v)^2} \leq \delta_{\max}^v$ at any angle, where $\delta_{\max}^u, \delta_{\max}^v$ are the maximum distance a UGV/UAV can travel given the fixed duration of a timeslot, and $\boldsymbol{x}_t^v = (x_t^v, y_t^v)$

is the position of a UAV v at the beginning of this timeslot. Each UAV v will consume $\eta\delta_t^v$ unit energy for movement, where η is a weight. The released UAVs will return to the designated UGV as UAV carrier after t^{rls} timeslots and fully charged to its initial energy level e_0^v, or before t^{rls} timeslots if it runs out of battery.

At each timeslot $[t, t+1)$, if a sensor p is within the sensing range of a UAV v, the latter collects $\triangle d_t^{v,p} = \min(\triangle d^{\mathcal{V}}, d_t^p)$ unit data, where $\triangle d^{\mathcal{V}}$ denotes the maximum amount of data each UAV v can collect from a sensor during each timeslot, and d_t^p denotes the remaining amount of data at sensor p at t.

B. Problem Definition

In air-ground collaborative SC tasks, UGVs and UAVs cooperate to collect data from deployed sensors within a sensing range, to achieve the following goals. First is to maximize the total amount of collected data in the workzone, defined as data collection ratio ψ:

$$\psi = 1 - \frac{\sum_p d_T^p}{\sum_p d_0^p}, \tag{3}$$

where $\sum_p d_0^p$ represents the initial data to be collected of all sensors in the target workzone, and $\sum_p d_T^p$ denotes the remaining data which UAVs fail to collect at the end.

Second is to try to collect data as evenly as possible among all sensors, defined as fairness ξ by by using Jain's fairness index [40] to measure the geographically exploration range of UGVs and UAVs, as:

$$\xi = \frac{\left(\sum_p \left(d_0^p - d_T^p\right)/d_0^p\right)^2}{P\sum_p \left(\left(d_0^p - d_T^p\right)/d_0^p\right)^2 + \epsilon}, \tag{4}$$

where ϵ is a small constant. Higher fairness indicates higher geographical coverage.

Third is to maximize the degree of UAVs-UGV air-ground collaboration since it is highly expected that UAVs can successfully collect sensory data every time it is released by its carrier UGV, defined as cooperation factor ζ:

$$\zeta = \frac{\sum_v \#_v^{\text{eff\_rls}}}{\sum_v \#_v^{\text{rls}}}, \tag{5}$$

where $\#_v^{\text{rls}}$ denotes the total number of UAV v is released in the task duration, $\#_v^{\text{eff\_rls}}$ counts the number of times a UAV v is released and actually collect data during this flight.

Fourth is to minimize the energy consumption of all UAVs due to constant movement, defined as energy consumption ratio β:

$$\beta = \frac{\sum_{v,t} \eta\delta_t^v}{\sum_v e_0^v + \sum_{v,t} \triangle e_t^v}, \tag{6}$$

where $\sum_{v,t} \triangle e_t^v$ denotes the overall charged energy for all UAVs.

Our goal is to maximize data collection ratio ψ, fairness ξ, cooperation factor ζ simultaneously, while minimizing energy consumption ratio β. Each of these metrics is a dimensionless scalar within the range of $[0, 1]$, which reflects part of the overall performance, and thus it is intuitive to multiply these metrics together to generate one single integrated metric as an overall performance index, called "efficiency" λ as:

$$\lambda = \frac{\psi \cdot \xi \cdot \zeta}{\beta}. \tag{7}$$

C. Problem Formulation

The state $\boldsymbol{s}_t \in \mathcal{S}$ of an air-ground SC task with UAV carriers contains four aspects. First is the information about the UGV stops and buildings as obstacles in the target workzone. Here we use a vector $\boldsymbol{x}_t^b$ to describe a UGV stop node b in stop graph $\mathcal{G}$, as:

$$\boldsymbol{x}_t^b = [x^b, y^b, d_t^b]^\top, \forall b \in \mathcal{B}, \tag{8}$$

where (x^b, y^b) denotes the position of stop node b, and d_t^b represents the data quantity that can be collected by UAVs released by UGV in the position of stop node b at t. Second is the current position $\boldsymbol{x}_t^u = (x_t^u, y_t^u)$ of each UGV u. Third is the current position $\boldsymbol{x}_t^v = (x_t^v, y_t^v)$ of each UAV v. Finally is the position $\boldsymbol{x}_p = (x_p, y_p)$ and the amount of remaining data d_t^p to be collected from each sensor p.

Each UGV u has its own observation $\boldsymbol{o}_t^u$, which contains two tensors. The first tensor $\hat{\boldsymbol{X}}_t^{\mathcal{B},u}$ records the observation of stop nodes, as:

$$\hat{\boldsymbol{X}}_t^{\mathcal{B},u} = [\hat{\boldsymbol{x}}_t^{1,u}, ..., \hat{\boldsymbol{x}}_t^{b,u}, ..., \hat{\boldsymbol{x}}_t^{B,u}]^\top, \tag{9a}$$

$$\hat{\boldsymbol{x}}_t^{b,u} = [x^b, y^b, \hat{d}_t^{b,u}]^\top, \tag{9b}$$

where $\hat{\boldsymbol{x}_t^b}$ is a masked vector of $\boldsymbol{x}_t^b$. Specifically, given a stop node b that has not been approached by the UAVs on UGV u until t, a constant is used to mask the value d_t^b in vector $\boldsymbol{x}_t^b$; and for a previously visited stop node b, the newest information $d_{t'}^p$ about the stop node b that approached at t' is used to mask the value d_t^b in vector $\boldsymbol{x}_t^b$. The second tensor $\boldsymbol{X}_t^{\mathcal{U}}$ records the observation of UGVs, denoted as:

$$\boldsymbol{X}_t^{\mathcal{U}} = [\boldsymbol{x}_t^1, ..., \boldsymbol{x}_t^u, ..., \boldsymbol{x}_t^U]^\top, \tag{10a}$$

$$\boldsymbol{x}_t^u = [x_t^u, y_t^u]^\top, \tag{10b}$$

where $\boldsymbol{x}_t^u$ denotes the position of a UGV u at t. The observation of a UAV v is then represented by:

$$\boldsymbol{o}_t^v = \boldsymbol{s}_t(x_t^v - l : x_t^v + l, y_t^v - l : y_t^v + l), \tag{11}$$

where (x_t^v, y_t^v) is the position of a UAV v, and l is a constant that controls the range of local observation.

The action $\boldsymbol{a}_t^u$ of a UGV u consists of two parts: $\boldsymbol{a}_t^u = (\omega, b_{tar})$, where in each timeslot a UGV u first decides whether to release the carried UAVs or not (denoted as ω), and if yes, let b_{tar} be the stop node to reach next. A UAV v's action $\boldsymbol{a}_t^v \in \mathbb{R}^2$ specifies its movement behavior in a 2D plate.

The reward function for a UGV u is as follows:

$$r_t^u = \begin{cases} 0, & \omega = \text{False} \\ \sum_{t'=t}^{t+t^{\text{rls}}} (\sum_{v \in \mathcal{V}^u} \triangle d_{t'}^v). & \text{otherwise} \end{cases} \tag{12}$$

If a UGV u decides not to release the carried UAVs and then take further movement to the next stop, it receives zero reward;

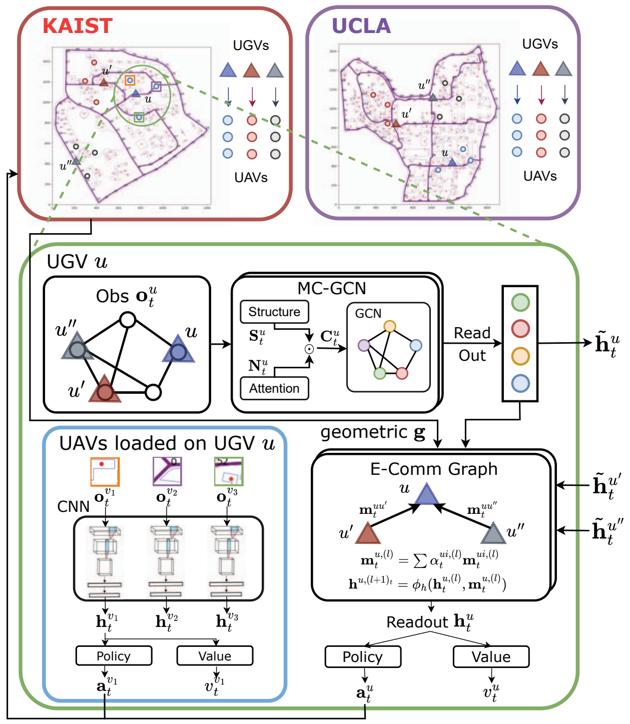


Fig. 2. Overall proposed model GARL.

otherwise, the data collected by the carried UAVs during $[t, t+t^{\text{rls}})$ will be returned as reward. The reward function for a UAV v is defined as $r_t^v = r_t^{v+} + r_t^{v-}$, where:

$$r_t^{v+} = \text{clip}\left(\frac{\xi_t \triangle d_t^v}{\eta \delta_t^v + \epsilon}, 0, \epsilon_3\right), \tag{13a}$$

$$\xi_t = \frac{\left(\sum_p \left(d_0^p - d_t^p\right)/d_0^p\right)^2}{P\sum_p \left(\left(d_0^p - d_t^p\right)/d_0^p\right)^2 + \epsilon}. \tag{13b}$$

Here, r_t^{u-} denotes the penalty when a UAV v crashes into obstacles, and ϵ_3, ϵ are both constants.

IV. Proposed Solution: GARL

A. Overview

As shown in Fig. 2, our proposed model GARL mainly consists of a multi-center attention-based GCN "MC-GCN" for UGV specific feature extraction, and a novel GNN-based communication mechanism "E-Comm" for MADRL that allows the UGV cooperations to be adaptive to the constant changing of geometric shapes formed by UGVs. For each UGV $u \in \mathcal{U}$, its structure can be depicted as follows:

$$\widetilde{\boldsymbol{h}}_t^u = \text{MC-GCN}(\boldsymbol{o}_t^u), \tag{14a}$$

$$\boldsymbol{h}_t^u = \text{E-Comm}(\widetilde{\boldsymbol{h}}_t^u, \{\widetilde{\boldsymbol{h}}_t^{u'}\}_{u' \in \mathcal{N}(u)}), \tag{14b}$$

$$\pi_{\boldsymbol{\theta}^u}(\boldsymbol{a}_t^u | \boldsymbol{h}_t^u) = f_u^{\pi}(\boldsymbol{h}_t^u), \tag{14c}$$

$$V_{\boldsymbol{\theta}^u}(\boldsymbol{h}_t^u) = f_u^V(\boldsymbol{h}_t^u), \tag{14d}$$

where $\mathcal{N}(u)$ denotes the set which contains the nearby UGV u' closing to a UGV u; $\boldsymbol{\theta}^u$ contains the total learnable parameters; $\pi_{\boldsymbol{\theta}^u}$ represents policy, and $V_{\boldsymbol{\theta}^u}$ represents value function. First, according to Eqn. (14a), MC-GCN takes the current observation $\boldsymbol{o}_t^u$ of a UGV u as input, and extracts UGV specific features $\widetilde{\boldsymbol{h}}_t^u$ from UGV stop network. Then, E-Comm module combines the UGV specific features and its neighboring UGVs to support geometrically equivalent cooperation and generates compact feature $\boldsymbol{h}_t^u$ for UGV u (see Eqn. (14b)). Subsequently, through $f_u^{\pi}(\cdot)$ and $f_u^V(\cdot)$, the compact feature $\boldsymbol{h}_t^u$ is mapped to an action distribution $\pi_{\boldsymbol{\theta}^u}(\boldsymbol{a}_t^u|\boldsymbol{h}_t^u)$ and a value function $V_{\boldsymbol{\theta}^u}(\boldsymbol{h}_t^u)$, respectively, as shown in Eqn. (14c) and (14d).

The policy function $\pi_{\boldsymbol{\theta}^u}$ is trained in the IPPO manner, as:

$$\begin{aligned} L_t^{CLIP}(\boldsymbol{\theta}^u) = \min\Big(&\frac{\pi_{\boldsymbol{\theta}^u}(\boldsymbol{a}_t^u|\boldsymbol{h}_t^u)}{\pi_{\boldsymbol{\theta}^u_{old}}(\boldsymbol{a}_t^u|\boldsymbol{h}_t^u)}A_t^u, \\ &\text{clip}\Big(\frac{\pi_{\boldsymbol{\theta}^u}(\boldsymbol{a}_t^u|\boldsymbol{h}_t^u)}{\pi_{\boldsymbol{\theta}^u_{old}}(\boldsymbol{a}_t^u|\boldsymbol{h}_t^u)}, 1-\epsilon_1, 1+\epsilon_1\Big)A_t^u\Big), \end{aligned} \tag{15}$$

where advantage function A_t^u is calculated via the Generalized Advantage Estimation (GAE) approach [41] and ϵ_1 is a constant. The value function $V_{\boldsymbol{\theta}^u}(\boldsymbol{h}_t^u)$ is used to generate the advantage function A_t^u, and its loss function is as:

$$\begin{aligned} L_t^{VF}(\boldsymbol{\theta}^u) = \max((V_{\boldsymbol{\theta}^u}(\boldsymbol{h}_t^u) - \hat{R}_t^u)^2, (\text{clip}(V_{\boldsymbol{\theta}^u}(\boldsymbol{h}_t^u), \\ V_{\boldsymbol{\theta}^u_{old}}(\boldsymbol{h}_t^u) - \epsilon_2, V_{\boldsymbol{\theta}^u_{old}}(\boldsymbol{h}_t^u) + \epsilon_2) - \hat{R}_t^u)^2, \end{aligned} \tag{16}$$

where $\hat{R}_t^u$ is the discounted reward; $V_{\boldsymbol{\theta}^u_{old}}(\boldsymbol{h}_t^u)$ is the value output in previous training steps; ϵ_2 is a constant.

Correspondingly, for each UAV $v \in \mathcal{V}$, the whole process can be described as:

$$\boldsymbol{h}_t^v = \phi_v(\boldsymbol{o}_t^v), \tag{17a}$$

$$\pi_{\boldsymbol{\theta}^v}(\boldsymbol{a}_t^v|\boldsymbol{h}_t^v) = f_v^{\pi}(\boldsymbol{h}_t^v), \tag{17b}$$

$$V_{\boldsymbol{\theta}^v}(\boldsymbol{h}_t^v) = f_v^V(\boldsymbol{h}_t^v), \tag{17c}$$

where $\phi_v(\cdot)$ contains several CNN layers and outputs a compact feature $\boldsymbol{h}_t^v$ for UAV v. Then, as shown in Eqn. (17b) and Eqn. (17c), the compact feature is mapped to an action distribution $\pi_{\boldsymbol{\theta}^v}(\boldsymbol{a}_t^v|\boldsymbol{h}_t^v)$ and a value function $V_{\boldsymbol{\theta}^v}(\boldsymbol{h}_t^v)$. Similarly, a UAV's policy is trained in the IPPO manner as well.

B. UGV Feature Extraction for Spatial Modeling by MC-GCN

In our considered air-ground tasks with UGVs, the size of an area to be explored by each UGV is limited if compared to the scope of the entire UGV stop network. Also, the data associated with PoIs may constantly change, making it quite challenging for UGVs to fully understand the entire environmental conditions in real-time. As a result, it is more likely for a UGV to randomly choose a stop node nearby based on its more precise local observations. To improve cooperation among UGVs, one possible solution is to use attention network but how to select an appropriate stop node by assigning different attention scores to the stop nodes becomes an issue. To solve this, each UGV can be seen as a positive center in the stop network, with stop nodes closer to it being assigned higher attention scores. The surrounding UGVs can be viewed as negative centers in the stop network, with stop nodes closer to them being assigned lower attention scores. By using these multiple centers, it becomes easier for each UGV to choose an appropriate stop node, preventing UGVs from staying in a small area and thus their cooperation is improved.

To this end, we propose MC-GCN, a multi-center attention-based GCN to extract UGV specific features from the global stop network. MC-GCN considers not only the position of each UGV itself, but also the positions of its nearby UGVs, to generate more valuable features of stop nodes. As a result, each UGV knows the position of other UGVs and can avoid competing for limited potential resources. This helps them to have a higher probability of exploring the unknown zone.

The forward process of MC-GCN can be divided into two phases. First, in Feature Collection Phase, MC-GCN captures two aspects of features (i.e., structure- and node-related features) from the stop network observed by each UGV to obtain graph as intact as possible. Specifically, each UGV maintains its own perceived global stop graph based on their observations, and MC-GCN process is conducted on each graph, respectively. From the perspective of a UGV u, the process of capturing structure-related features can be depicted by:

$$\hat{s}(b_t^u, b) = s\left(b_t^u, b\right) - \frac{1}{U-1} \sum_{u' \in \mathcal{U}-\{u\}} s(b_t^{u'}, b), \tag{18a}$$

$$\boldsymbol{S}_t^u = \left[\hat{s}(b_t^u, 1), ..., \hat{s}(b_t^u, b), ..., \hat{s}(b_t^u, B)\right]^\top, \tag{18b}$$

where $s(\cdot)$ represents a structural correlation function that captures the graphical structural relationship between two nodes (e.g., the reciprocal of shortest path distance), and b_t^u denotes the current stop node of a UGV u at t.

According to Eqn. (18a), $\hat{s}(b_t^u, b)$ can be regarded as structural relevance evaluation of stop node b by UGV u, where the relationship between other UGVs and node b are also taken into account. The structural relevance $s(b_t^{u'}, b)$ of other UGVs on node b is then subtracted to consider the possibility that a stop node b may be accessed by other UGVs. Finally, we combine the structural information from all nodes to get structure-related features $\boldsymbol{S}_t^u$ of a UGV u. Specifically, we introduce shortest path distance [42] between nodes b_m and b_n, as:

$$d_{sp}^q = \begin{cases} d_{sp}(b_m, b_n), & d_{sp}(b_m, b_n) \leq q \\ \infty & \text{otherwise}, \end{cases} \tag{19}$$

where threshold q defines the scope of relevance between nodes. If two nodes whose shortest path distance exceeds q, they are considered as unreachable nodes from each other. MC-GCN uses the reciprocal of the distance, meaning that closer nodes have stronger relationship in graph structure. To avoid the interference of zero denominator, we add a small positive term to the denominator as:

$$s(b_t^u, b) = \frac{1}{d_{sp}^q(b_t^u, b) + 1}. \tag{20}$$

To this end, the structure-related feature $\boldsymbol{S}_t^u$ of UGV u's observed graph is obtained.

Next, in Feature Extraction Phase, MC-GCN utilizes the previously obtained features to generate the weights of GCN layers, and eventually, obtains the UGV specific feature representation through the GCN with attention module. It guides each UGV to focus more on those stop nodes which not only have relatively more remaining data (thus collecting them will generate reward), but also far away ones to avoid possible interest overlap. Specifically, the structural-related feature $\boldsymbol{S}$ and node-related feature $\boldsymbol{N}$ are combined to measure the importance of UGV stop nodes by attention mechanism, as:

$$\boldsymbol{F}_t^{uu',(l)} = \boldsymbol{H}_t^{u,(l)} \boldsymbol{W}_1^{(l)} (\boldsymbol{H}_t^{u,(l)}[b_t^{u'}])^\top, \tag{21a}$$

$$\boldsymbol{N}_t^{u,(l)} = \boldsymbol{F}_t^{uu,(l)} - \frac{1}{U-1} \sum_{u' \in \mathcal{U}-\{u\}} \boldsymbol{F}_t^{uu',(l)}, \tag{21b}$$

$$\boldsymbol{C}_t^{u,(l)} = \text{Softmax}(\boldsymbol{S}_t^u \cdot \boldsymbol{N}_t^{u,(l)}), \tag{21c}$$

where $\boldsymbol{F}_t^{uu',(l)}$ is the output of attention module in l-th GCN layer with attention reduction and $\boldsymbol{W}_1$ is a learnable weight matrix. We use node $b_t^{u'}$ to attend on all other nodes with a UGV in the graph from its perception. Then, similar to structural-related feature, the node-related feature is obtained by subtracting self-node attention from cross-node attention. This kind of operation allows a UGV u to find the important nodes to focus on and keep the interest separated from other UGVs, by considering others' attention on these nodes. Finally, the attention weight $\boldsymbol{C}$ is combined with these two features by softmax normalization.

The mechanism above is operated in each GCN layer, providing additional weights in the forward process of GCN:

$$\boldsymbol{H}_t^{u,(l+1)} = \sigma(\boldsymbol{C}_t^{u,(l)} \cdot (\boldsymbol{L}\boldsymbol{H}_t^{u,(l)}\boldsymbol{W}_2^{(l)})), \tag{22}$$

where $\boldsymbol{L}$ is the Laplace matrix and $\boldsymbol{H}_t^{u,(l)}$ is UGV u's graph input for the l-th layer. For the initial setting, we take $\boldsymbol{H}_t^{u,(0)} = \hat{\boldsymbol{X}}_t^{\mathcal{B},u}$ from a UGV's observation. Finally, the UGV specific features $\widetilde{\boldsymbol{h}}_t^u$ of stop network centered on a UGV u can be extracted from the top layer of GCN outputs:

$$\widetilde{\boldsymbol{h}}_t^u = \phi_H(\boldsymbol{H}_t^{u,(-1)}), \tag{23}$$

where ϕ_H is a linear readout function.

C. Equivariant Multi-Agent Communication among UGVs by E-Comm

Existing communication mechanisms, which are typically based on GNN models that are invariant to translation and rotation of the geometric attributes of graph nodes (in our case, UGVs) [16], [17], are unable to adapt to the constant changes in geometric shapes formed by the UGVs. This is problematic in complex environments where these geometric changes should be taken into consideration, particularly in situations where the shape formed by the UGVs may remain similar after a geometric change, but the UGVs themselves must behave differently due to changes in the associated left-over data.

Inspired by Equivariant Graph Neural Networks (EGNNs [16]), we propose E-Comm module, which is a GNN-based communication mechanism that well adapts to constant changing of geometric shapes formed by UGVs, including translation and rotation. In E-Comm, we regard each UGV as a graph node to build a communication graph, and design an equivariant message passing scheme among UGVs, as part of proposed decentralized MADRL framework. Through carefully leveraging the geometric features of UGVs (the coordinates of UGVs), E-Comm allows UGVs to be aware of the geometric changes and adapt the process of message exchange.

In detail, E-Comm consists of invariant *Message Aggregation phase*, and equivariant *Target Updating phase*, then the composition of them still preserves equivariance. In forward process, we introduce non-geometric feature $h_t^{u,(l)}$ and geometric feature $g_t^{u,(l)}$ into E-Comm layers, which are initialized as:

$$\boldsymbol{h}_t^{u,(0)} = \widetilde{\boldsymbol{h}}_t^u, \tag{24a}$$

$$\boldsymbol{g}_t^{u,(0)} = \boldsymbol{x}_t^u. \tag{24b}$$

These two features are messages transmitted among UGVs and updated through E-Comm layers.

1) Message Aggregation Phase among UGVs: To aggregate messages from other nodes, E-Comm is operated in a weighted-sum manner, where the weights are designed based on the geometric distance among UGVs. We estimate the distance between UGV u and u' by their geometric feature, as:

$$\boldsymbol{r}_t^{uu',(l)} = \boldsymbol{g}_t^{u,(l)} - \boldsymbol{g}_t^{u',(l)}, u' \in \mathcal{N}(u), \tag{25}$$

where $\mathcal{N}(u)$ represents the neighborhood of a UGV u. Since this substraction only preserves the relative position between two UGVs, the output is geometrically invariant.

In the communication graph, distant UGVs tend to affect less with each other. For a UGV u, we take reciprocal of the norms $\|\boldsymbol{r}_t^{uu',(l)}\|$ to generate the normalized importance weights $\alpha_t^{uu',(l)}$ of messages from a UGV u', given $u' \in \mathcal{N}(u)$, as:

$$\alpha_t^{uu',(l)} = \frac{\exp(\|\boldsymbol{r}_t^{uu',(l)}\|^{-1})}{\sum_{u''\in\mathcal{N}(u)} \exp(\|\boldsymbol{r}_t^{uu'',(l)}\|^{-1})}. \tag{26}$$

With the weights, the message aggregation process in l-th E-Comm layer can be defined by:

$$\boldsymbol{m}_t^{uu',(l)} = \phi_m^{(l)}(\boldsymbol{h}_t^{u',(l)}), \tag{27a}$$

$$\boldsymbol{m}_t^{u,(l)} = \sum_{u'\in\mathcal{N}(u)} \alpha_t^{uu',(l)} \boldsymbol{m}_t^{uu',(l)}, \tag{27b}$$

$$\boldsymbol{h}_t^{u,(l+1)} = \phi_h^{(l)}([\boldsymbol{h}_t^{u,(l)}; \boldsymbol{m}_t^{u,(l)}]), \tag{27c}$$

where ϕ_m and ϕ_h are linear functions. From the view point of a UGV u, the aggregated message $\boldsymbol{m}_t^{u,(l)}$ combines message $\boldsymbol{m}_t^{uu',(l)}$ from other UGVs in its neighborhood. Finally, UGV u's non-geometric feature is updated by the aggregated message for next layer.

2) Target Updating Phase among UGVs: In each E-Comm layer, the geometric feature $\boldsymbol{g}_t^{u,(l)}$ is updated in a radial direction guided by the distance measurement $\boldsymbol{r}_t^{uu',(l)}$ to estimate the target position of a UGV u. It is notable that the measurement $\boldsymbol{r}_t^{uu',(l)}$ contains directional information since this measurement is a subtraction of geometric feature of a pair of UGVs which are learned from coordination in 2D space.

Here, we introduce $\tilde{\boldsymbol{g}}_t^{u,(l)}$ to estimate the joint effect on a UGV u from those UGVs in u's neighborhood:

$$\tilde{\boldsymbol{g}}_t^{u,(l)} = \sum_{u'\in\mathcal{N}(u)} \alpha_t^{uu',(l)} \phi_g^{(l)}(\boldsymbol{m}_t^{uu',(l)}) \hat{\boldsymbol{r}}_t^{uu',(l)}, \tag{28}$$

where $\phi_g^{(l)}$ is a linear message encoder, $\hat{\boldsymbol{r}}_t^{uu',(l)}$ is the unit vector of $\boldsymbol{r}_t^{uu',(l)}$ to keep directional information only. Referring to resultant force in physics, the vector sum of $\hat{\boldsymbol{r}}_t^{uu',(l)}$ accumulate the effect from others, which tends to keep a UGV u from gathering with other UGVs. Considering this effect, the geometric feature of UGV u is updated by:

$$\boldsymbol{g}_t^{u,(l+1)} = \boldsymbol{g}_t^{u,(l)} + \mathrm{clip}(\tilde{\boldsymbol{g}}_t^{u,(l)}, \tilde{\boldsymbol{g}}_{max}). \tag{29}$$

To constrain the update range, $\tilde{\boldsymbol{g}}_t^{u,(l)}$ is clipped by a constant vector $\tilde{\boldsymbol{g}}_{max}$.

Finally, we readout the features from the top layer of E-Comm to extract the graph representation. This process in depicted by:

$$\boldsymbol{z}_t^u = \hat{\boldsymbol{X}}_t^{\mathcal{B},u}[:2] \boldsymbol{W}_3 (\boldsymbol{g}_t^{u,(-1)})^\top, \tag{30a}$$

$$\boldsymbol{h}_t^u = \phi_u([\boldsymbol{h}_t^{u,(-1)}; \boldsymbol{z}_t^u]), \tag{30b}$$

where $\boldsymbol{z}_t^u$ extracts the relationship between the position of target $\boldsymbol{g}_t^{u,(-1)}$ and all stop nodes, which reflects UGV u's preference among all stops.

As shown in Eqn. (30b), invariant feature $\boldsymbol{h}_t^{u,(-1)}$ and equivariant feature $\boldsymbol{z}_t^u$ compose the final readout, which is geometrically equivariant as well.

D. Algorithm Description and Computational Complexity Analysis

The entire training process of GARL is shown in Algorithm 1. We train GARL for M iterations. First, we initialize learnable parameters for UGVs and UAVs, respectively (Line 1), and start the loop for sampling and training (Line 2). Since the base design our model is IPPO, we initialize the training buffer $\mathcal{D}^u$ for a UGV u, and the training buffer $\mathcal{D}^v$ for a UAV v (Line 3). Next, we start the loop for sampling in the air-ground SC environment, where the duration of a task is divided into T timeslots (Line 4). Then, all UGVs interact with the environment together (Line 5). In each timeslot $[t, t+1)$, each UGV u first gets local observation $\boldsymbol{o}_t^u$ from the environment (Line 6). Then, a UGV u generates its specific feature $\boldsymbol{h}_t^u$ using Eqn. (14a) and Eqn. (14b) (Line 7). After, it samples its own action $\boldsymbol{a}_t^u$ according to $\pi_{\boldsymbol{\theta}^u}(\boldsymbol{a}_t^u|\boldsymbol{h}_t^u)$ (Line 8), and calculates value function V_t^u according to $V_{\boldsymbol{\theta}^u}(\boldsymbol{h}_t^u)$ (Line 9). If a UGV u decides to release the loaded UAVs following its sampled action, or a UGV u is still waiting for the UAVs to return, all

Algorithm 1: GARL

```
1  Initialize UGV parameter $\boldsymbol{\theta}^u$ and UAV parameter $\boldsymbol{\theta}^v$.
2  for iteration$= 1, 2, \cdots, M$ do
3      Set training buffer $\mathcal{D}^u = \mathcal{D}^v = \{\}$;
4      for $t = 1, 2, \cdots, T$ do
5          for $u = 1, 2, \cdots, U$ do
6              Get local observation $\boldsymbol{o}_t^u$;
7              Generate feature $\boldsymbol{h}_t^u$ by Eqn. (14a) and
                Eqn. (14b);
8              Sample action $\boldsymbol{a}_t^u \sim \pi_{\boldsymbol{\theta}^u}(\boldsymbol{a}_t^u|\boldsymbol{h}_t^u)$;
9              Calculate value $V_t^u \leftarrow V_{\boldsymbol{\theta}^u}(\boldsymbol{h}_t^u)$;
10             foreach UAV $v$ for $u$ do
11                 Get local observation $\boldsymbol{o}_t^v$;
12                 Generate feature $\boldsymbol{h}_t^v$ by Eqn. (17);
13                 Sample action $\boldsymbol{a}_t^v \sim \pi_{\boldsymbol{\theta}^v}(\boldsymbol{a}_t^v|\boldsymbol{h}_t^v)$;
14                 Calculate value $V_t^v \leftarrow V_{\boldsymbol{\theta}^v}(\boldsymbol{h}_t^v)$;
15         Execute actions of all UGVs and UAVs,
            receive reward and transit to next state;
16     foreach UGV and UAV do
17         Compute accumulative rewards $\hat{R}_{1:T}$ and
            advantage $A_{1:T}$ from trajectory;
18         if UGV then
19             $\mathcal{D}^u = \mathcal{D}^u \cup \{(\hat{R}_{1:T}, A_{1:T}, V_{1:T})\}$;
20         else
21             $\mathcal{D}^v = \mathcal{D}^v \cup \{(\hat{R}_{1:T}, A_{1:T}, V_{1:T})\}$;
22     Optimize surrogate loss in Eqn. (2) w.r.t $\boldsymbol{\theta}^u$ and
        $\boldsymbol{\theta}^v$, with $J$ times and mini-batch sampled from
        $\mathcal{D}^u$ and $\mathcal{D}^v$;
23     $\boldsymbol{\theta}_{old}^u \leftarrow \boldsymbol{\theta}^u$, $\boldsymbol{\theta}_{old}^v \leftarrow \boldsymbol{\theta}^v$.
```

UAVs released by this UGV will interact with the environment together (line 10). Each UAV v first gets its local observation $\boldsymbol{o}_t^v$ from the environment (Line 11). Then, a UAV v extracts its distinct feature $\boldsymbol{h}_t^v$ using Eqn. (17) (Line 12). After, it samples its own action $\boldsymbol{a}_t^v$ according to $\pi_{\boldsymbol{\theta}^v}(\boldsymbol{a}_t^v|\boldsymbol{h}_t^v)$ (Line 13), and calculate value function V_t^v according to $V_{\boldsymbol{\theta}^v}(\boldsymbol{h}_t^v)$ (Line 14). After all UGVs and UAVs execute their actions, they will receive an individual reward and the environment transits to next state (Line 15). Then, we start to train the parameters of the neural network models for UGVs and UAVs (Line 16). We compute accumulative rewards $\hat{R}_{1:T}^u$ and advantage $A_{1:T}^u$ from trajectory for each UGV and UAV (Line 17). For a UGV (Line 18), we add the tuple $(\hat{R}_{1:T}, A_{1:T}, V_{1:T})$ into training buffer $\mathcal{D}^u$ (line 19); otherwise (Line 20), the tuple will be added into training buffer $\mathcal{D}^v$ (Line 21). After collecting the training samples, we start the training process. We optimize $\boldsymbol{\theta}^u$ and $\boldsymbol{\theta}^v$ by Eqn. (2) in total J times through sampling mini-batches from $\mathcal{D}^u$ and $\mathcal{D}^v$, respectively (Line 22-23).

GARL includes the vanilla PPO policy network that contains several convolution and linear layers. We compute the time complexity of forward process according to [43]. For UGVs, MC-GCN, E-Comm and PPO network handle vector input, which only contain fully connected layers, as:

$$O\Big(\sum_{i=1}^{H_L} D_{1,i} \cdot D_{2,i}\Big), \tag{31}$$

where H_L is the number of linear layers; $D_{1,i}$ and $D_{2,i}$ are the dimension of input and output features of i-th linear layers.

For UAVs, they take image input as local observation, thus both convolution layers and linear layers are included in the forward process. The time complexity for UAVs can be computed by:

$$O\Big(\sum_{i=1}^{H_L} D_{1,i} \cdot D_{2,i} + \sum_{i=1}^{H_C} D_{3,i}^2 \cdot D_{4,i}^2 \cdot D_{5,i} \cdot D_{6,i}\Big), \tag{32}$$

where H_L and H_C is the number of linear layers and convolution layers; $D_{1,i}$ and $D_{2,i}$ are the dimension of input and output features of i-th linear layers; $D_{3,i}$, $D_{4,i}$, $D_{5,i}$ and $D_{6,i}$ are the size of output feature maps and convolution kernel, the number of input channels and output channels of the i-th convolution layer.

V. Performance Evaluation

A. Campus Description and Simulation Setting

We use two campuses KAIST, South Korea and UCLA, USA, to simulate our considered air-ground SC task with UAV carriers, and in particular to collect CCTV camera and sensory data deployed in/around the buildings. Campus lanscapte data are obtained from Google Map by using OpenStreetMap and pre-processed for our simulation, including setting the boundaries of buildings and the trace of roads. KAIST campus has a relatively simpler road network, while UCLA campus is more complicated. Google Map is used to mark the positions and the shapes of buildings and mountains. KAIST spans 1433.37 meters from north to south and 1539.63 meters from east to west, covering about 2.21 million square meters. We randomly placed 138 sensors on 85 buildings; UCLA spans 1737.15 from north to south and 1675.36 from east to west, covering 2.91 million square meters. Likewise, we place 236 sensors on 163 buildings in UCLA campus.

UAVs have weak long-distance travelling capability due to limited battery supply but good short-distance mobility. However, UGVs have good long-distance mobility but can only move following the roads. We set the length of each timeslot as 30 seconds. Each sensor p is initialized with data amount d_0^p, which is randomly generated within range 1GB to 1.5GB. The initial location of each UGV is set in the center of two campuses. According to TS-X4, the maximum flying speed of UAVs is 12km/h and its initial energy reserve is e_0 = 10kJ [44]. Energy consumption weight factor η = 0.01kJ/m, the sensing range for UAVs is 60 meters, and the data collection rate for UAVs are 166.7 Mbps per sensor, respectively. For UGVs, we put UGV stops every 100 meters along the roads and assume that UGVs can maximally travel 400 meters in a timeslot (equivalent to the maximum speed 48km/h [45]).

TABLE II
IMPACT OF NO. OF LAYERS IN MC-GCN L^{MC} AND NO. OF LAYERS IN E-COMM L^E (WHEN $U = 4, V' = 2$).

		L^{MC}					L^E				
		1	2	3	4	5	1	2	3	4	5
KAIST	λ	0.8280	0.9211	**0.9970**	0.9760	0.8665	0.7215	0.9064	**0.9970**	0.9852	0.9487
	ψ	0.5023	0.4221	0.6198	0.5234	0.5236	0.5394	0.5552	0.6198	0.6156	0.5937
	ξ	0.5282	0.4439	0.6391	0.5513	0.5475	0.5558	0.5716	0.6391	0.6323	0.6209
	ζ	0.7025	0.7676	0.6760	0.7277	0.6588	0.5445	0.5174	0.6760	0.5783	0.6270
	β	0.2361	0.1663	0.2786	0.2228	0.2255	0.2533	0.1913	0.2786	0.2493	0.2539
UCLA	λ	0.5190	0.5619	**0.6137**	0.5961	0.5658	0.5099	0.5548	**0.6137**	0.5804	0.5645
	ψ	0.3600	0.3634	0.4511	0.3897	0.3891	0.3510	0.3716	0.4511	0.3730	0.3725
	ξ	0.3812	0.3818	0.4667	0.4121	0.4155	0.3723	0.3885	0.4667	0.3981	0.3951
	ζ	0.7230	0.7346	0.7244	0.7215	0.7435	0.7005	0.6913	0.7244	0.7703	0.7422
	β	0.1985	0.1882	0.2613	0.2024	0.2193	0.1939	0.2000	0.2613	0.2109	0.2059

TABLE III
ABLATION STUDY ($U = 4$ AND $V' = 2$).

Campus	Method	λ	ψ	ξ	ζ	β
KAIST	GARL	**0.9970**	0.6198	0.6391	0.6760	0.2786
	GARL w/o MC	0.7036	0.4952	0.5205	0.6575	0.2530
	GARL w/o E	0.8119	0.5303	0.5548	0.6760	0.2573
	GARL w/o MC, E	0.5810	0.4478	0.4742	0.6269	0.2470
UCLA	GARL	**0.6137**	0.4511	0.4667	0.7244	0.2613
	GARL w/o MC	0.4114	0.3553	0.3799	0.7039	0.2426
	GARL w/o E	0.5080	0.3721	0.3898	0.7163	0.2123
	GARL w/o MC, E	0.3396	0.3200	0.3343	0.7033	0.2356

In all the experiments, we use Pytorch 1.8.1 to implement our proposed model, and all the codes are run on Ubuntu 18.04.2 LTS with 8 GeForce RTX A6000 graphic cards. We conduct three sets of experiments, including impact of hyperparameters, ablation study and comparing with baselines. Results are compared from data collection ratio ψ, fairness ξ, cooperation factor ζ and low energy consumption ratio β, as well as the final performance index efficiency λ.

B. Impact of Hyperparameters

We select two key hyperparameters from MC-GCN, where L^{MC}, L^E determines the number of layers in MC-GCN, E-Comm, respectively. As shown in Table II, we find that all of two hyperparameters yield a peak value in terms of efficiency. When L^{MC} is too small, the receptive field of each UGV stop node during graph convolution will not be big enough to consider sufficient neighboring nodes' features; in another extremity, when the L^{MC} is too big, the features extracted from each UGV stop node are too general since they consider too many nodes which are far away. Obviously, either extremity worsens the overall performance. Similarly, in E-Comm, when the L^E is too small, each UGV can only obtain the information of UGVs nearby, which will result in low level of cooperative policies for the UAVs-UGV coalition. On the other hand, when the L^E is too big, the received message may contain redundant information which makes it difficult for a UGV to extract useful ones from the message. Therefore, we choose $L^{MC} = 3$ and $L^E = 3$ as the best hyperparameters used in the following experiments.

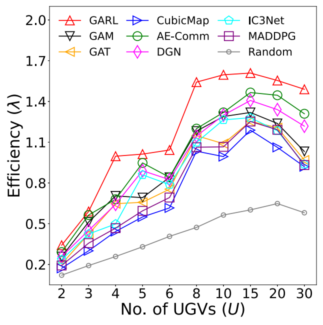
C. Ablation Study

We choose 4 UGVs and 2 UAVs per UGV as the setting of our ablation study, which is performed by gradually removing two key components of our model, i.e., MC-GCN (MC) and E-Comm (E). As shown in Table III, the complete model GARL achieves 41.70% and 49.17% higher efficiency than that of GARL w/o MC in KAIST and UCLA campuses, respectively. This confirms that proposed MC-GCN model is able to extract feature from the state more comprehensively especially when the environment (i.e., UCLA campus) is more complicated (compared to KAIST campus). If not, GCN employs only graph convolution, missing the different importance between neighboring nodes generated by attention mechanism as in our proposed MC-GCN. Thus the feature extraction may attach more importance on nodes far away, which will lead the UGVs to go to certain places with less data to collect. GARL achieves 22.81% and 20.79% higher efficiency than that of GARL w/o E in two campuses, due to the help of our proposed communication mechanism between UGVs in message integration. This benefit is clear for UCLA campus whose topographic landscape is more irregular and its east/west parts are connected with a thin area which does not contain much data to be collected. Furthermore, GARL w/o MC achieves 21.09% and 21.16% higher efficiency than that of GARL w/o MC, E in two campuses. Finally, GARL w/o E achieves 39.72% and 49.61% higher efficiency than that of GARL w/o MC, E in two campuses, which proves that our proposed MC-GCN model do help UGVs accurately locate the most important information relevant to the current UGV's partial observation.

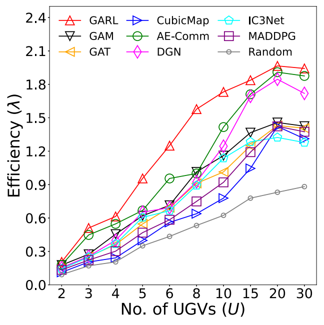
D. Comparing with Eight Baselines

We compare our method GARL with eight baselines, as:

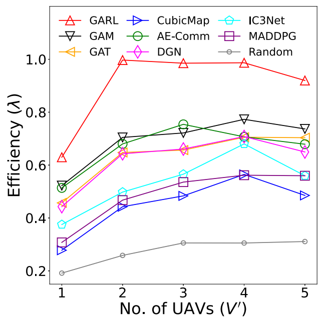
- **CubicMap** [11]: It is a memory augmented CNN-based method with cubic writing and spatially contextual reading mechanisms to extract long-term spatiotemporal features. We consider it as the state-of-the-art approach for UAV-aided SC.
- **GAM** [14]: It is a GNN based method that adaptively accesses the sequence of UGV stop nodes ordered by the importance to extract both long-term and short-term spatiotemporal features. We consider it as the state-of-the-art approach for spatial modeling.
- **GAT** [13]: It is a classical graph feature extraction method as an extension of GNN where attention mechanism is used to attach importance on different neighboring nodes.
- **AE-Comm** [46]: It designs a communication auto-encoder to generate a common language among all MADRL agents. This auto-encoder is used to transform the agent's observation to the common language representation. We consider it as the state-of-the-art approach for communication based MADRL methods.
- **DGN** [47]: It is an attention based method to enhance agent communication that attaches different importance on neighboring nodes and aggregates their messages according to their importance.
- **IC3Net** [15]: It is a classical communication model for multi-agent environment which uses individualized LSTM policy and a gating mechanism to control when to communicate among agents.



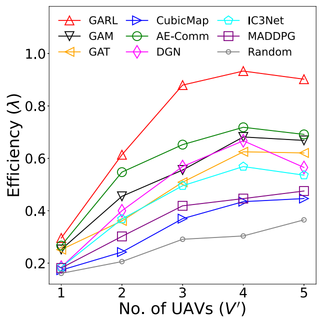
(a) KAIST ($V' = 2$)



(b) UCLA ($V' = 2$)

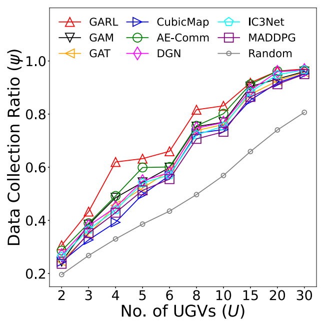


(c) KAIST ($U = 4$)

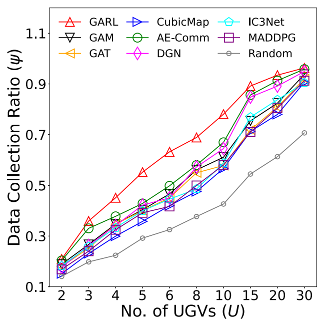


(d) UCLA ($U = 4$)

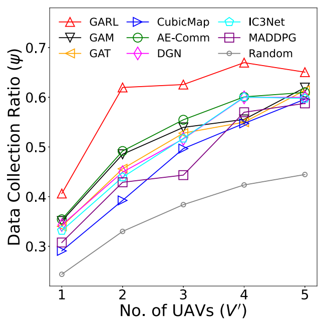
Fig. 3. Impact of no. of UGVs and UAVs in terms of efficiency λ.



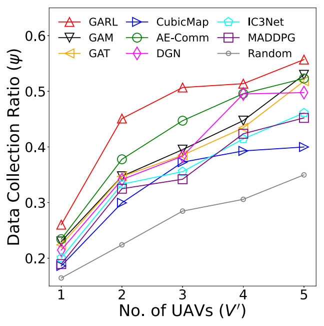
(a) KAIST ($V' = 2$)



(b) UCLA ($V' = 2$)



(c) KAIST ($U = 4$)



(d) UCLA ($U = 4$)

Fig. 4. Impact of no. of UGVs and UAVs in terms of data collection ratio ψ.

- **MADDPG** [36]: It is a classical MADRL method aided by a distributed prioritized experience replay.
- **Random**: For each UGV and UAV, we sample an action uniformly from action space.

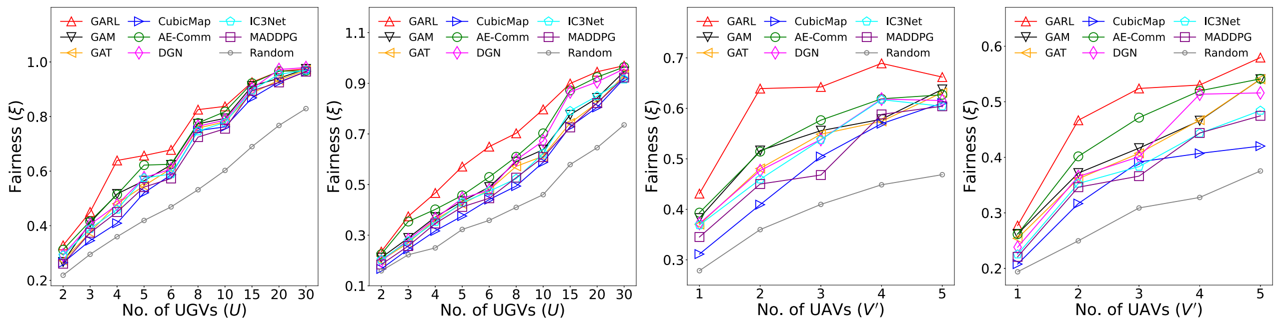
Results are shown in Fig. 3 - Fig. 6. We make four important observations.

Our proposed model GARL consistently outperforms all eight baselines in terms of efficiency in both KAIST and UCLA campus environments. The reason comes from three aspects, namely: accurate feature extraction from spatial modeling, efficient communication mechanism among UAGs, and used underlying MADRL base model. In terms of spatial modeling, CubicMap is a memory augmented CNN-based method, but not based on GNN, thus cannot have clear overview of the geometric structure of the UGV stop network, which is crucial in our considered air-ground SC tasks with UAV carriers. GAT uses attention mechanism to capture the different degrees of importance of immediate neighboring UGV stop nodes, however it does not consider all other UGV stop nodes which can be a bit far away but still useful to understand the current entire workzone state. GAM slightly outperforms GAT, due to its combination with GNN and LSTM that traverses all the neighboring stop nodes. Since both methods are built on the view of single UGV, they can not distinguish the benefits brought by other UGVs, thus it is likely that UGVs may gather together around certain stop nodes with high importance without exploring the whole workzone, resulting in poor fairness, data collection ratio, and ultimately overall efficiency.

In terms of communication mechanism among MADRL agents, IC3Net employs LSTM to aggregate the messages received from other UGVs during several past timeslots. This helps learn from historical actions, however simply using average operation to compute target estimation is not enough which blurs the distinct geometric feature of neighboring UGVs. DGN uses attention mechanism to evaluate the importance of messages from neighboring UGVs. However, it does not fully consider the constant geometrical changes of the shape formed by UGVs into the design of MADRL communication mechanism, which is quite essential especially when sensory data distribution is not uniform. AE-Comm outperforms DGN and IC3Net by a large margin. It transforms local geometric feature into a global language for all the UGVs to understand, so the message aggregation phase will be much easier and comprehensive. Unfortunately, AE-Comm lacks a mechanism to carefully handle the spatial information; as a result, its performance is worse than our methods.

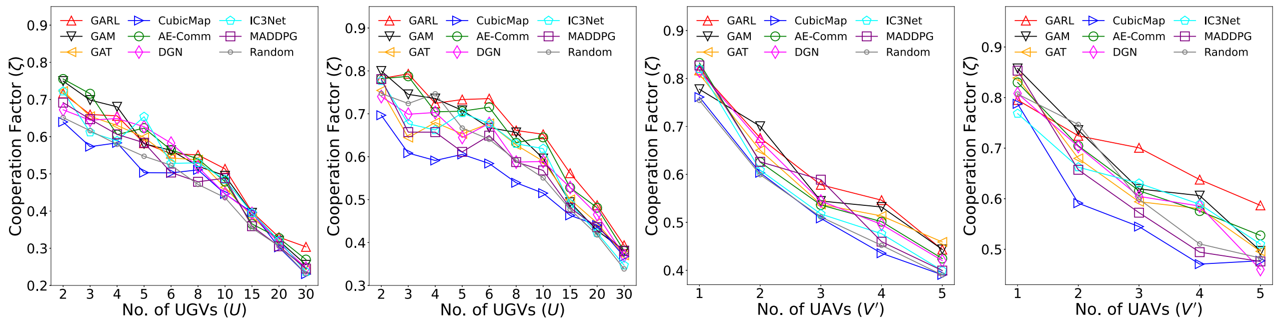
In terms of used underlying MADRL base model, although MADDPG is quite classical, its employed deterministic policy DDPG is not good at action exploration, which is crucial in our campus environment. We use IPPO as the start point of our design, which has much better capability to explore the complex environment than MADDPG.

The spatial complexity of two campus environments has



(a) KAIST ($V' = 2$) (b) UCLA ($V' = 2$) (c) KAIST ($U = 4$) (d) UCLA ($U = 4$)

Fig. 5. Impact of no. of UGVs and UAVs in terms of fairness ξ.



(a) KAIST ($V' = 2$) (b) UCLA ($V' = 2$) (c) KAIST ($U = 4$) (d) UCLA ($U = 4$)

Fig. 6. Impact of no. of UGVs and UAVs in terms of cooperation factor ζ.

big impact on the performance of all methods. In GARL, we see that UCLA cannot obtain the same level of efficiency as KAIST, when fewer UGVs and UAVs are employed. However, as more UGVs are added (thus loaded UAVs as coalitions), UCLA receives much higher efficiency than KAIST. This is because that not much data can be found in the center of UCLA campus (fewer building as lawns), and in order to collect data in the west, UAVs need to be carried by UGVs, possibly from east to the west. This observation is consistent for other spatial modeling baselines, that performance under KAIST is better than UCLA.

When increasing the number of UGVs, the attained efficiency from all methods first goes up and then down, as shown in Fig. 3(a) and Fig. 3(b). In detail, when U changes from 2 to 15 in KAIST and from 2 to 20 in UCLA dataset, efficiency increases because more UGV-UAVs coalitions are employed to collect more data and cover wider areas, resulting in rapid increase of both data collection ratio and geographic fairness. However, the cooperation factors of all methods decrease (see Fig. 6(a) and Fig. 6(b)). This is because the competition among UAVs becomes fierce, and the probability of collecting data when a UAV is released is also reduced. We also notice that when U changes from 15 to 30 in KAIST and from 20 to 30 in UCLA dataset, overall efficiency starts to decrease. This is because too many UGVs will not bring further benefits to the data collection ratio and fairness, while the cooperation factor will drop dramatically due to the large number of ineffective UAV flights.

When increasing the number of UAVs loaded on each UGV, Random approach does not change much for all metrics because it lacks enough explorations or spatial modeling. The attained efficiency from other methods first goes up then down, as shown in Fig. 3(c) and Fig. 3(d) in KAIST. This is because that there may not be enough data to collect around a single UGV thus UAVs released by this UGV will waste time searching in the same area. As a result, we see data collection ratio goes up but both fairness and cooperation factor go down. In UCLA, efficiency will go up rapidly from $V' = 1$ to $V' = 4$ (see in Fig. 3(d)), and then fall down when $V' = 5$, so the bottleneck becomes the number of UGV-UAVs coalition for range extension. In terms of cooperation factor, all methods give downward trend (see Fig. 6(c) and Fig. 6(d)). This is because that since all the UAVs in a particular UGV are released at the same point, competition is inevitable, leaving an insight that too many UAVs on one single UGV may not be helpful to bring further benefits.

E. Illustrative Trajectories of UGV-UAVs Coalitions

In Fig. 7, we show the trace of UGV-UAVs coalitions after running 100 timeslots when $U = 4, V' = 2$. We choose four best baselines (two communication based methods and two spatial modeling methods), AE-Comm, DGN, GAM and GAT,

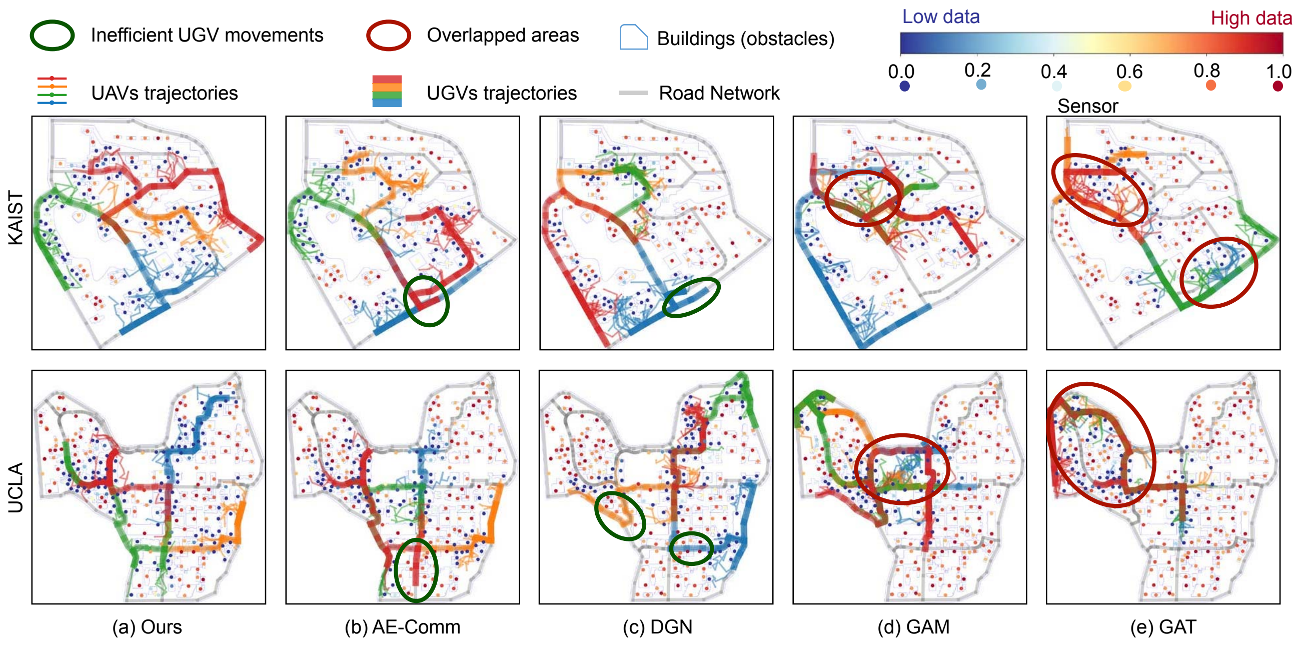


Fig. 7. Movement trace of UGVs and UAVs in KAIST and UCLA (when $U = 4$ and $V' = 2$).

for illustration, according to Section V-D, as well as GARL. We see that AE-Comm and DGN lead to meaningless UGV wanderings in the same areas multiple times (see inefficient UGV movement in KAIST, from Fig. 7(b) and Fig. 7(c)), or simply carrying UAVs to move around even if there is adequate data to be collected (see inefficient UGV movement in UCLA, from Fig. 7(b) and Fig. 7(c)). However, this phenomenon is not likely to happen in GAM or GAT, but may lead to competitive UGV gathering in the same area (see overlapped areas in Fig. 7(d) and Fig. 7(e)). On the contrary, our method GARL produces nice UGV trajectories to be responsible for a sub-workzone (no overlapping or missing data movements), to release the UAVs to collect the data nearby, before heading to the next UGV stop.

F. Computational Complexity Analysis

Computational complexity in terms of both time cost and graphic card memory usage during testing phase is shown in Table IV. With $U = 4$ and $V' = 2$, we observe the running time for a UGV from inputting observation to producing actions in a timeslot by our method GARL. The time cost is slightly higher than some baselines. However, it is still in the scale of millisecond, which is negligible in practice. Even if our proposed model GARL uses relatively high amount of graphic card memory compared with MADDPG, it is still acceptable compared with the benefits it brings, which has been shown in previous sections.

TABLE IV
COMPUTATIONAL COMPLEXITY OF ALL METHODS.

Method	Time Cost (ms)		Graphic Card Mem. Usage (MB)	
	KAIST	UCLA	KAIST	UCLA
GARL	0.553	1.121	935	937
GAM [14]	0.66	1.167	939	945
GAT [13]	0.493	0.552	813	841
CubicMap [11]	1.023	2.417	1348	1506
AE-Comm [46]	0.552	0.786	907	943
DGN [47]	**0.379**	**0.523**	935	937
IC3Net [15]	0.688	0.892	975	997
MADDPG [36]	2.108	3.892	**805**	**836**

VI. Conclusion

In this paper, we explicitly consider a new type of air-ground SC tasks with UAV carriers, and navigate a group of UGV-UAVs coalitions to perform sensory data collection. Specifically, we propose a novel deep model called GARL, which consists of a spatial modeling module MC-GCN (to extract UGV specific feature from its own observation of UGV stop network), and E-Comm for MADRL communication (equivariant to geometric transformation of the shape formed by UGVs considering uneven distribution of data over the environment). We conduct extensive experiments on two campus environments KAIST and UCLA, where results confirm that our proposed method consistently outperforms all eight baselines in terms of efficiency.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No. U21A20519 and 62022017). Ye Yuan is supported by the National Key Research and Development Program of China (Grant No. 2022YFB2702100) and the NSFC (Grant Nos. 61932004, 62225203, U21A20516). Corresponding Author: Chi Harold Liu.

References

[1] C. H. Liu, J. Fan, J. W. Branch, and K. K. Leung, "Toward qoi and energy-efficiency in internet-of-things sensory environments," *IEEE Transactions on Emerging Topics in Computing*, vol. 2, no. 4, pp. 473–487, 2014.

[2] C. H. Liu, B. Zhang, X. Su, J. Ma, W. Wang, and K. K. Leung, "Energy-aware participant selection for smartphone-enabled mobile crowd sensing," *IEEE Systems Journal*, vol. 11, no. 3, pp. 1435–1446, 2015.

[3] B. Zhang, C. H. Liu, J. Tang, Z. Xu, J. Ma, and W. Wang, "Learning-based energy-efficient data collection by unmanned vehicles in smart cities," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 4, pp. 1666–1676, 2017.

[4] C. H. Liu, Y. Zhao, Z. Dai, Y. Yuan, G. Wang, D. Wu, and K. K. Leung, "Curiosity-driven energy-efficient worker scheduling in vehicular crowdsourcing: A deep reinforcement learning approach," in *IEEE ICDE'20*, 2020, pp. 25–36.

[5] Z. Chen, P. Cheng, L. Chen, X. Lin, and C. Shahabi, "Fair task assignment in spatial crowdsourcing," *Proc. VLDB Endow.*, vol. 13, no. 11, pp. 2479–2492, 2020. [Online]. Available: http://www.vldb.org/pvldb/vol13/p2479-chen.pdf

[6] M. Samir, S. Sharafeddine, C. M. Assi, T. M. Nguyen, and A. Ghrayeb, "Uav trajectory planning for data collection from time-constrained iot devices," *IEEE Transactions on Wireless Communications*, vol. 19, no. 1, pp. 34–46, 2020.

[7] C. H. Liu, Z. Chen, and Y. Zhan, "Energy-efficient distributed mobile crowd sensing: A deep learning approach," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1262–1276, 2019.

[8] 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 J. Sel. Areas Commun.*, vol. 36, no. 9, pp. 2059–2070, 2018.

[9] C. H. Liu, Z. Dai, Y. Zhao, J. Crowcroft, D. Wu, and K. K. Leung, "Distributed and energy-efficient mobile crowdsensing with charging stations by deep reinforcement learning," *IEEE Trans. Mob. Comput.*, vol. 20, no. 1, pp. 130–146, 2021.

[10] H. Wang, C. H. Liu, Z. Dai, J. Tang, and G. Wang, "Energy-efficient 3d vehicular crowdsourcing for disaster response by distributed deep reinforcement learning," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pp. 3679–3687.

[11] Y. Wang, C. H. Liu, C. Piao, Y. Yuan, R. Han, G. Wang, and J. Tang, "Human-drone collaborative spatial crowdsourcing by memory-augmented and distributed multi-agent deep reinforcement learning," in *IEEE ICDE'22*, 2022, pp. 459–471.

[12] T. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *ArXiv*, vol. abs/1609.02907, 2017.

[13] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[14] A. Wijesinghe and Q. Wang, "A new perspective on" how graph neural networks go beyond weisfeiler-lehman?"," in *ICLR'21*, 2021.

[15] A. Singh, T. Jain, and S. Sukhbaatar, "Learning when to communicate at scale in multiagent cooperative and competitive tasks," in *ICLR'18*, 2018.

[16] V. G. Satorras, E. Hoogeboom, and M. Welling, "E (n) equivariant graph neural networks," in *ICML'21*, 2021, pp. 9323–9332.

[17] J. Han, Y. Rong, T. Xu, and W. Huang, "Geometrically equivariant graph neural networks: A survey," *arXiv preprint arXiv:2202.07230*, 2022.

[18] S. Sukhbaatar, R. Fergus *et al.*, "Learning multiagent communication with backpropagation," *NIPS'16*, vol. 29, 2016.

[19] C. S. de Witt, T. Gupta, D. Makoviichuk, V. Makoviychuk, P. H. Torr, M. Sun, and S. Whiteson, "Is independent learning all you need in the starcraft multi-agent challenge?" *arXiv preprint arXiv:2011.09533*, 2020.

[20] C. H. Liu, Q. Lin, and S. Wen, "Blockchain-enabled data collection and sharing for industrial iot with deep reinforcement learning," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3516–3526, 2018.

[21] L. Kazemi and C. Shahabi, "Geocrowd: enabling query answering with spatial crowdsourcing," in *Proceedings of the 20th international conference on advances in geographic information systems*, 2012, pp. 189–198.

[22] W. Ni, P. Cheng, L. Chen, and X. Lin, "Task allocation in dependency-aware spatial crowdsourcing," in *IEEE ICDE'20*, 2020, pp. 985–996.

[23] P. Cheng, L. Chen, and J. Ye, "Cooperation-aware task assignment in spatial crowdsourcing," in *IEEE ICDE'19*, 2019, pp. 1442–1453.

[24] Z. Zhou, J. Feng, B. Gu, B. Ai, S. Mumtaz, J. Rodriguez, and M. Guizani, "When mobile crowd sensing meets UAV: energy-efficient task assignment and route planning," *IEEE Trans. Commun.*, vol. 66, no. 11, pp. 5526–5538, 2018. [Online]. Available: https://doi.org/10.1109/TCOMM.2018.2857461

[25] 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, 2020.

[26] J. Zhang, Z. Yu, S. Mao, S. C. G. Periaswamy, J. Patton, and X. Xia, "Iadrl: Imitation augmented deep reinforcement learning enabled ugv-uav coalition for tasking in complex environments," *IEEE Access*, vol. 8, pp. 102 335–102 347, 2020.

[27] M. L. e. a. Wang Y, Wang M, "New crowd sensing computing in space-air-ground integrated networks," *International Conference on Space-Air-Ground Computing (SAGC)*, vol. 7, 2021.

[28] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. v. d. Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in *European semantic web conference*. Springer, 2018, pp. 593–607.

[29] R. Li, S. Wang, F. Zhu, and J. Huang, "Adaptive graph convolutional neural networks," in *AAAI'18*, vol. 32, no. 1, 2018.

[30] L. Wu, P. Sun, R. Hong, Y. Fu, X. Wang, and M. Wang, "Socialgcn: An efficient graph convolutional network based model for social recommendation," *arXiv preprint arXiv:1811.02815*, 2018.

[31] M. Sakai, K. Nagayasu, N. Shibui, C. Andoh, K. Takayama, H. Shirakawa, and S. Kaneko, "Prediction of pharmacological activities from chemical structures with graph convolutional neural networks," *Scientific reports*, vol. 11, no. 1, pp. 1–14, 2021.

[32] L. Bai, L. Yao, C. Li, X. Wang, and C. Wang, "Adaptive graph convolutional recurrent network for traffic forecasting," *NeurIPS'20*, vol. 33, pp. 17 804–17 815, 2020.

[33] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *ICML'17*, 2017, pp. 1263–1272.

[34] F. A. Oliehoek and C. Amato, *A concise introduction to decentralized POMDPs*. Springer, 2016.

[35] S. Li, C. H. Liu, Q. Lin, Q. Wen, L. Su, G. Huang, and Z. Ding, "Deep residual correction network for partial domain adaptation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 7, pp. 2329–2344, 2020.

[36] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *NIPS'17*, vol. 30, 2017.

[37] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in *AAAI'18*, vol. 32, no. 1, 2018.

[38] 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 *ICML'18*, 2018, pp. 4295–4304.

[39] C. Yu, A. Velu, E. Vinitsky, Y. Wang, A. Bayen, and Y. Wu, "The surprising effectiveness of ppo in cooperative, multi-agent games," *arXiv preprint arXiv:2103.01955*, 2021.

[40] R. K. Jain, D.-M. W. Chiu, W. R. Hawe *et al.*, "A quantitative measure of fairness and discrimination," *Eastern Research Laboratory, Digital Equipment Corporation, Hudson, MA*, 1984.

[41] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," *arXiv preprint arXiv:1506.02438*, 2015.

[42] A. Madkour, W. G. Aref, F. U. Rehman, M. A. Rahman, and S. Basalamah, "A survey of shortest-path algorithms," *arXiv preprint arXiv:1705.02044*, 2017.

[43] K. He and J. Sun, "Convolutional neural networks at constrained time cost," in *IEEE CVPR'15*, 2015, pp. 5353–5360.

[44] "TS-X4," http://www.droneyee.com/droneDetail/138.

[45] "D80," https://auto.dji.com/cn/solution-d80.

[46] T. Lin, J. Huh, C. Stauffer, S. N. Lim, and P. Isola, "Learning to ground multi-agent communication with autoencoders," *NeurIPS'21*, vol. 34, pp. 15 230–15 242, 2021.

[47] J. Jiang, C. Dun, T. Huang, and Z. Lu, "Graph convolutional reinforcement learning," in *ICLR'19*, 2019.