# Learning to Coordinate for a *Worker-Station* Multi-robot System in Planar Coverage Tasks

Jingtao Tang[1,2], Yuan Gao[2], Tin Lun Lam[1,2,†]

***Abstract*— For massive large-scale tasks, a multi-robot system (MRS) can effectively improve efficiency by utilizing each robot's different capabilities, mobility, and functionality. In this paper, we focus on the multi-robot coverage path planning (mCPP) problem in large-scale planar areas with random dynamic interferers in the environment, where the robots have limited resources. We introduce a *worker-station* MRS consisting of multiple *workers* with limited resources for actual work, and one *station* with enough resources for resource replenishment. We aim to solve the mCPP problem for the *worker-station* MRS by formulating it as a fully cooperative multi-agent reinforcement learning problem. Then we propose an end-to-end decentralized online planning method, which simultaneously solves coverage planning for *workers* and rendezvous planning for *station*. Our method manages to reduce the influence of random dynamic interferers on planning, while the robots can avoid collisions with them. We conduct simulation and real robot experiments, and the comparison results show that our method has competitive performance in solving the mCPP problem for *worker-station* MRS in metric of task finish time.**

## I. Introduction

For massive large-scale tasks in hazardous environments, Multi-Robot System (MRS) dramatically helps to reduce human exposure to potential dangers and improves efficiency effectively. There are various applications of MRS that have come to reality, including search and rescue [1], persistent surveillance [2], planetary exploration [3]. Typically, a robot has only limited working resources, including energy and consumables. For example, most robots are driven by electrical or thermal energy stored in batteries or fuels in advance. While some robots can obtain ambient energy from the environment (e.g., solar energy), the energy transfer is highly dependent on the environmental situation, and the recharging process can sometimes be slow. Thus it can be inefficient for robots in massive long-term tasks. In scenarios like cleaning or agriculture on large-scale fields, the robot has limited consumables like water or chemicals. Therefore, it is essential for robots with large-scale tasks to constantly travel between supply stations and working areas to replenish and work, which is very inefficient for such tasks.

As discussed by Vaughan et al. in [4], the placement of the supply station significantly influences work efficiency for an MRS in the above scenarios. To further improve the efficiency of the MRS, one might consider making the supply station a mobile robot platform. Similar to the *Frugal Feeding Problem* in [5], the station moves around to serve the working robots. For consistency in this paper, we name such an MRS as the "*worker-station*" MRS, which is composed of a mobile supply station robot and several working robots. We consider the Multi-robot Coverage Path Planning (mCPP) problem on planar areas for the aforementioned *worker-station* MRS. As shown in Fig. 1, the *workers* are equipped with a range device for general area coverage work, and the *station* is loaded with sufficient resources to provide supplies for *workers*. The joint objective is to cover a given target area as soon as possible. Solving such a planning problem for the *worker-station* MRS can be decomposed as below:

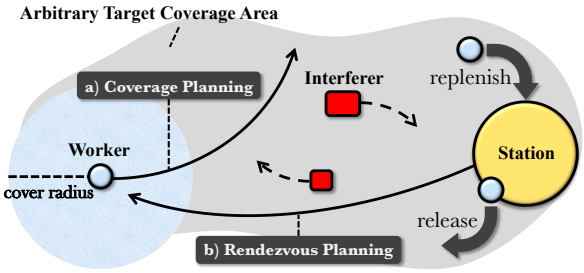



Fig. 1: Given an arbitrary target coverage area, a mCPP problem for the *worker-station* MRS on planar areas can be decomposed into: 1) coverage planning for *workers* (blue robots) and 2) rendezvous planning for *station* (yellow robot). There are random dynamic interferers (red robots) in the environment.

1) Coverage planning for each *worker* to finish general planar coverage work of a given area;
2) Rendezvous planning for the *station* to service *workers* in need of replenishment;

In this paper, we mainly focus on solving the mCPP problem for the *worker-station* MRS on planar areas. There are several challenges to the above problem. First, the joint problem space comprised of the above two planning problems is too large to solve directly and simultaneously. A practical solution is to discretize state and action spaces in mCPP problems [6] and rendezvous planning problems [7], then solve by discrete combinatorial optimization methods separately. However, the system dynamics are hard to model and identify, where each robot has different capabilities and functionality. Thus, such methods can still be infeasible for such complex MRS, even after reducing the problem size. Secondly, planning with dynamic collision avoidance is another challenge for most offline planning methods. One general solution is to combine offline planning with

[1]Authors are with School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen. Authors' emails are todd.j.tang@gmail.com and tllam@cuhk.edu.cn.

[2]Authors are with Shenzhen Institute of Artificial Intelligence and Robotics for Society, Shenzhen, Guang Dong, China. Author's email is gaoyuankidult@gmail.com

We gratefully acknowledge support of the National Key R&D Program of China (2020YFB1313300) and the funding (AC01202101103) from the Shenzhen Institute of Artificial Intelligence and Robotics for Society.

[†]Corresponding author

local collision avoidance controllers. Nevertheless, such a hierarchical planning scheme would alter the optimal policy that is planned offline without the interference of dynamic obstacles. For complex scheduling tasks like the mCPP problem for *worker-station* MRS, it will cause conflicts and even deadlocks between robots or planners, and the planning efficiency will get worse as the number of robots grows [8]. To tackle the above challenges, we adopt Deep Reinforcement Learning (DRL) to solve the mCPP problem for *worker-station* MRS. However, the coordination behaviors of different agents in the *worker-station* MRS are nontrivial to learn together, and agents often struggle between exploration and exploitation of the coverage task during training. We summarize the main contributions of this paper below:

1) We propose an end-to-end decentralized online planning method to solve the mCPP problem for the *worker-station* MRS. Our method manages to reduce the influence of random dynamic interferers on planning, while the robots can avoid collisions with them.
2) We design a two-stage curriculum learning with an intrinsic curiosity module and soft approximation of the *workers*' energy constraints, which successfully guide the training for large-scale coverage tasks.
3) We provide ablation study, simulation, and real robot experimental results. The results show that our method outperforms decomposition-based and graph-based baseline methods in coverage finish time metrics.

## II. Related Work

### A. Multi-robot Coverage Path Planning

The mCPP problem evolved from the classical Coverage Path Planning (CPP) problem by introducing multiple robots to solve the coverage problem. Most approaches are based on the graph structure, which is proven to be NP-hard [9]. Zheng *et al.* designed a constant-factor approximation algorithm in polynomial time [10]. Kapoutsis *et al.* uses an area division algorithm to allocate tasks for multiple robots [11]. Apart from graph-based methods, decomposition-based methods also take large parts in the literature [12], [13], which first partition the target area into obstacle-free convex sub-regions for different robots and then apply single robot coverage planning for each robot separately. Most graph-based or decomposition-based mCPP methods do offline planning, and some also require the coverage area to satisfy specific assumptions (e.g., convex-shaped area). In addition, classic offline mCPP methods can be undermined by random dynamic interferers in the environment.

On the other hand, recently, some works have been extending the mCPP problem to various applications with specific constraints, such as geophysical surveys [14], fault-tolerant planning on large-scale outdoor [15]. However, to the best of our knowledge, there are few works on the mCPP problem for the aforementioned *worker-station* MRS.

### B. Worker-Station Multi-robot System

Similar to the *worker-station* MRS, related works on mobilizing the supply station into an autonomous robot mainly focus on rendezvous planning for *station* to efficiently recharge the *workers* in need. For example, Couture et al. [4] only plans for *station*, while *workers* are dedicated to delivering goods between fixed source and destination. Similarly, in [16], only rendezvous planning of *stations* is considered, whereas the *workers* are programmed to monitor the environment by predefined trajectories persistently. Most of these works consider the *workers* to be stationary in terms of their motion patterns and state transitions, which reduce the complexity to a solvable level for optimization.

More recently, Yu et al. [17] tried to solve both planning problems for one *worker* and one *station*, but it is restricted to node coverage for a given graph. Similarly in Sun et al. [18], the *worker* is planned to travel between waypoints, while the *station* is planned to rendezvous to charge the *worker*. Seyedi et al. [19] planned trajectories for multiple *workers* and one *station* with scheduled charging order, which also requires a prior of the environment. Despite the above work managed to plan for both *workers* and *station*, it is only applicable on convex target areas with static obstacles, thus is infeasible in an arbitrary target area with dynamic interferers.

## III. Problem Formulation

In this section, we provide our Multi-agent Reinforcement Learning (MARL) problem formulation of the Multi-robot Coverage Path Planning (mCPP) problem on planar areas, for the *worker-station* Multi-robot System (MRS) (see Fig. 1). Given target area $\Omega$, the *worker-station* MRS consists of $m$ *workers* $W^i$ and $n$ *stations* $S^j$, where $i = 1, 2, ..., m$ and $j = 1, 2, ..., n$. The goal is to find the optimal policy for each robot in the *worker-station* MRS, to minimize the coverage task finish time while avoiding collisions with dynamic interferers in the environment.

### A. Preliminaries

In this subsection, we first introduce several preliminary concepts and assumptions in the rest of the paper.

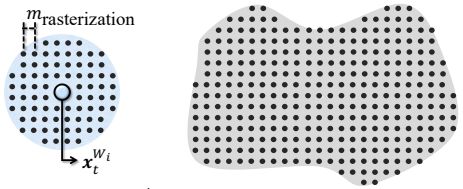
*1) Worker robot and station robot:* we consider both *workers* and *station* have limited range of perception and communication: within the perception range, each robot can detect collisions and objects precisely; within the communication range, each robot can receive information from other robots (e.g., the rough global position of other robots). As mentioned previously in the *worker-station* MRS, *workers* also have limited energy, while *station* have unlimited energy to replenish *workers*. Note that the coverage work range of *worker* does not necessarily equal its perception range.

*2) Energy Capacity and Rendezvous Recharge:* denote the energy capacity for *worker* $W^i$ as a constant $c^i$. Suppose the current energy left for $W^i$ at time $t$ is $e^i_t$, then current percentage of remained energy $p^i_t$ is defined as: $p^i_t = \frac{e^i_t}{c^i} \in [0, 1]$. A *worker* $W^i$ is said to be "*exhausted*" if $p^i_t$ is lower than a threshold $p^e$, otherwise it is said to be "*normal*". Also, since we mainly focus on the planning problem at a higher level in this paper, the local rendezvous of *workers* and *station* is simplified by comparing with a position threshold $\varepsilon$. We assume a *worker* can be replenished by any *station*, then the

discharge and recharge for each *worker* $W^i$ is determined by comparing $\varepsilon$ with the euclidean distance between the global positions of $\boldsymbol{x}_t^{W^i}$ and $\boldsymbol{x}_t^{S^j}$ for *worker* $W^i$ and *station* $S^j$:

$$e_t^i = \begin{cases} \max\{0,\ e_{t-1}^i - e_{\text{discharge}}\}, & \|\boldsymbol{x}_t^{W^i} - \boldsymbol{x}_t^{S^j}\| > \varepsilon \\ \min\{c^i,\ e_{t-1}^i + e_{\text{charge}}\}, & \|\boldsymbol{x}_t^{W^i} - \boldsymbol{x}_t^{S^j}\| \leq \varepsilon \end{cases} \quad (1)$$

*3) Coverage Task and Synchronized Coverage Area:* coverage by *workers* is only carried out when *worker* has energy left. Once a *worker* starts to recharge, it would be released from *station* if and only if it is fully recharged (i.e., $p_t^i = 1$). Denote the coverage area of each *worker* $W^i$ at time $t$ as $\mathcal{C}_t^i$. Here we assume the overall covered area $\mathcal{C}_t$ at time $t$ can be updated and synchronized among all agents during the task. The update and synchronization of $\mathcal{C}_t$ can be implemented by mutual information exchange for robots within the communication range: $\mathcal{C}_t = \bigcup_{t=0}^{T} \bigcup_{i=1}^{m} \mathcal{C}_t^i$.




(a) Coverage area $\mathcal{C}_t^i$ (b) Arbitrary target coverage area $\Omega$

Fig. 2: Model the coverage task by uniform sampling

For a released *worker* $W^i$ with energy left (i.e., $p_t^i > 0$), the coverage area $\mathcal{C}_t^i$ at time $t$ is determined by uniformly sampling as depicted in Fig. 2: given the sampling resolution $m_{\text{rasterization}}$, the coverage area $\mathcal{C}_t^i$ is approximated by uniformly sampling the coordinates within the shape boundaries of the target coverage area $\Omega$ (i.e., rasterization). Then, the coverage area is represented by a set of coordinates in planar space. Thus, the union operations on coverage areas $\mathcal{C}_t$ turn into set union operations, which is computationally tractable using a hash-set compared with area union operations.

### B. Multi-agent Reinforcement Learning

We first introduce the Decentralized Partially Observable Markov Decision Process (Dec-POMDP) [20] denoted by $(\mathcal{R}, \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{O}, r, b)$, where $\mathcal{R}$ is the set of agents, $\mathcal{S}$ is the joint state space, $\mathcal{A}$ is the joint action space, $\mathcal{P}$ is the state-transition model, $\mathcal{O}$ is joint observation space, $r$ is the shared reward function and $b$ is the initial state distribution. With a shared reward function $r$, we can formulate the MARL problem by single-agent reinforcement learning objective [21]:

$$\max_{\pi}\ \mathbb{E}\left[\sum_{t \leq T, s_0 \sim b} \gamma^t r\left(s_t, a_t, s_{t+1}\right) \mid a_t^i \sim \pi(\cdot \mid o_t^i)\right] \quad (2)$$

where $\pi$ is the policy and $\gamma$ is reward discount factor, $s_t \in \mathcal{S}$ and $a_t \in \mathcal{A}$ are the joint state and action of agents at time $t$ respectively. Given the initial state of agents $s_0$ and the observation $o_t^i$ at time $t$, the action $a_t^i$ is sampled from the policy $\pi$. The goal is to maximize the expected discounted reward within a time horizon of $T$. Note that Eq. 2 can be adopted for agents with different observations or functionalities (each corresponds to a different policy), as long as they share a common reward function.

### C. Worker-Station MRS Coverage Task Formulation

As introduced previously, the *worker-station* MRS consists of two types of agents with different functionality: the *workers* are responsible for coverage work with limited energy, whereas the *station* is responsible for replenishing *workers* with unlimited energy. Thus following Eq. 2, we define the agents $\mathcal{R} = \{W^i\}_{i=1}^m \bigcup \{S^j\}_{j=1}^n$ as the set of *stations* and *workers*, and $\mathcal{A} = \{a^i\}_{i=1}^{m+n}$ are the actions of *workers* and *stations* sequentially. We define policy $\pi_\phi$ and policy $\pi_\theta$ for *stations* and *workers* respectively, and a shared reward function $r$ for both agents. Then, we can formalize the mCPP problem for *worker-station* MRS as a fully cooperative MARL problem [22]:

$$\pi_\theta^*, \pi_\phi^* = \arg\max_{\pi_\theta, \pi_\phi}\ \mathbb{E}\left[\sum_{t \leq \min\{T, T_{\text{finish}}\}} \gamma^t r\left(s_t, a_t, s_{t+1}\right)\right] \quad (3)$$

where $a_t^i \sim \pi_\theta(\cdot \mid o_t^i), i = 1, 2, ..., m$ are the actions of *workers*, and $a_t^j \sim \pi_\phi(\cdot \mid o_t^j), j = m+1, m+2, ..., m+n$ are the actions of *stations*. Note that the planning horizon $T$ is replaced by the minimum of the original time horizon $T$ and the coverage task finish time $T_{\text{finish}}$.

In order to finish the cooperative coverage task as soon as possible, with collision avoidance and rendezvous to recharge, we define the shared reward function $r$ as below:

$$r = \sum_{i=0}^{k} ({}^c r)^i + \sum_{i=0}^{k} ({}^e r)^i + r_{\text{collision}} + r_{\text{time}} \quad (4)$$

The first component $({}^c r)^i$ is the covering reward for $i$-th *worker*, which is the only positive term to guide the coverage planning of *workers*. The second component $({}^e r)^i$ is a penalty term added when a *worker*'s energy is close to its energy capacity, which guides the rendezvous planning of *station*. The details of the first two reward components are elaborated in Sec. IV. The third component $r_{\text{collision}}$ is a constant collision penalty whenever a collision occurs, which guides the agents for dynamic collision avoidance. The last component $r_{\text{time}}$ is a constant time penalty in each time step whenever the coverage task has not finished, which helps to find more time-efficient planning policies.

Since the total coverage reward of the coverage area $\Omega$ is a constant (i.e., only new covered area provide rewards), and the other three penalty terms would stop accumulating once the coverage task finishes, only a time-optimal coverage and rendezvous planning policy with collision avoidance can reach the optimal task performance. Therefore, by designing and selecting appropriate rewards for the above components in Eq. 3, we can apply MARL algorithms to train the agents for the *worker-station* MRS coverage task to coordinate with each other for theoretical optimal performance.

## IV. Deep Reinforcement Learning Approach

In this section, we introduce key components of our DRL-based planning method. We follow the paradigm of centralized training and decentralized execution (CTDE) [23], which has been widely used in MARL for Dec-POMDP

modeled robot learning problems [24], [25]. The training and planning phases are elaborated in Sec. IV-A. In Fig. 4, we summarize our end-to-end planning pipeline. For an ego agent (*worker* or *station*), the perception-range and communication-range observations described in Sec. IV-B are encoded into feature vectors by a convolution neural network. Then, they are stacked upon zero-range observation, constituting the final observation vector $\hat{o}_t^i$ in latent space. The policy network $\pi_\theta$ for *worker* and $\pi_\phi$ for *station* are both Multi-layer Perceptron (MLP) modules, each of which takes $\hat{o}_t^i$ for the $i$-th agent at time $t$ and output its action $a_t^i$. The action $a_t^i$ is then converted to velocity commands as mentioned in Sec. IV-C, which solves both rendezvous planning for *station* and coverage planning for *workers*.

### A. DRL Training and Planning Phases

We follow the paradigm of CTDE to train the policy network of each agent in Eq. 3, then deploy the corresponding policy networks on robots for planning. During training, the state of the whole system and observations of all other agents are needed for better training performance in simulation. During planning, each agent receives only its zero-range, perception-range, and communication-range observations as described in Sec. IV-B, then output the best action according to the corresponding trained policy network. We unfold the details in CTDE in the following two parts.

*1) Centralized training phase:* for training algorithm, we use the multi-agent actor-critic algorithm MA-POCA [26] to train the policy networks for *workers* and *stations*. During training, a centralized critic network is trained to estimate the value of the current system state, including the whole system and observations of all agents. Note that states and observations of the whole system are only needed during training and can be easily accessed in simulations. According to [27], such a centralized critic network would greatly help the training of the actor network (i.e., policy network).

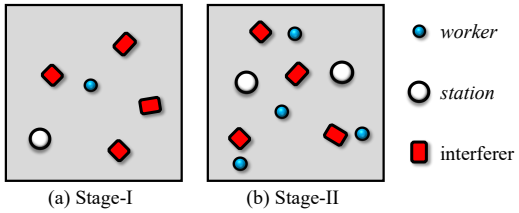



Fig. 3: Two-stage curriculum learning for mCPP problem of *worker-station* MRS: (a) Stage-I: one *station* with single *worker*; (b) Stage-II: multiple *stations* with multiple *workers*.

For better policy exploration of the coordination behaviors towards the coverage task during training, we adopt the Intrinsic Curiosity Module (ICM) [28]. In short, the ICM trains a self-supervised inverse dynamic model that predicts the consequences of an agent's actions, and uses that prediction error as an intrinsic reward to guide the agent's exploration during training. In considerations of training performance, we designed a two-stage curriculum learning [29] evolving from single *worker* into multiple *workers*, which guides *workers* and *station* for better policies during training.

As shown in Fig. 3-(a), stage-I is designed to make it easier for both *worker* and *station* to focus on learning some basic behaviors, such as the ability of collision avoidance with static obstacles and dynamic interferers. For *worker*, the "*cover and replenish*" behavior is learnt when the remained energy of *worker* is at a low level. For *station*, the behavior of finding and following exhausted *worker* is learnt. Once training of stage-I is converged, we can then extend the *worker* and *station* to multiple ones, and adapt the pre-trained policy networks to train for final policies (see Fig. 3-(b)).

*2) Decentralized execution phase:* unlike the centralized training phase, each agent only needs its own observation during the decentralized planning phase. Specifically speaking, each agent only takes its own observation as introduced in IV-B, and outputs optimal action by its observation and the corresponding policy network $\pi_\phi$ and $\pi_\theta$.

### B. Observation Space

For both *worker* and *station*, the observation $\hat{o}_t^i$ of the $i$-th ego agent at time $t$ consists of following three types: 1) zero-range observation $({}^z o)_t^i$ contains its own basic information; 2) perception-range observation $({}^p o)_t^i$ contains precise local information within its perception range; 3) communication-range observation $({}^c o)_t^i$ contains rough global information within its communication range. A demonstration of observation is shown in the ego agent observation block in Fig. 4.

| | **perception-range** | **communication-range** |
|---|---|---|
| ***worker*** | *worker* *station*<br>obstacle uncovered area | *worker* *station*<br>uncovered area |
| ***station*** | obstacle *worker*(normal)<br>*worker*(exhausted) | *station* *worker*(normal)<br>*worker*(exhausted) |

TABLE I: Encoded objects in perception-range and communication-range observations for ego agents in the *worker-station* MRS.

Here we elaborate on each type of observation for an ego agent. For both *workers* and *stations*, $({}^z o)_t^i$ includes global position and local velocity, which are then stacked vertically as 1-D zero-range observation. Note that when the $i$-th agent is *worker*, the percentage of remaining energy $p_t^i$ is also included in $({}^z o)_t^i$. Both $({}^p o)_t^i$ and $({}^c o)_t^i$ are encoded as images with object positions (see Fig. 4), which are translated and rotated with the $i$-th ego agent. The encoded objects in $({}^p o)_t^i$ and $({}^c o)_t^i$ are listed in Tab. I. For $({}^p o)_t^i$, it is a $20 \times 20$ image with $n_p$ channels (i.e., the number of encoded objects) and $m_{\text{perc}}$ grid resolution (i.e., length per pixel). For $({}^c o)_t^i$, it is a $30 \times 30$ image with $n_c$ channels and $m_{\text{comm}}$ grid resolution.

### C. Action Space

We define the action space as a 2d continuous vector space consisting of linear and angular velocities. Given the max linear velocity $v_{\text{max}}^i$ and max angular velocity $\omega_{\text{max}}^i$ of the $i$-th robot , the sampled action $a_t^i$ is scaled by multiplying $v_{\text{max}}^i$ or $\omega_{\text{max}}^i$ to give the desired velocity commands.

### D. Reward Design

As mentioned in Eq. 4, the shared reward $r$ for all agents consists of four components. Here we only elaborate on the

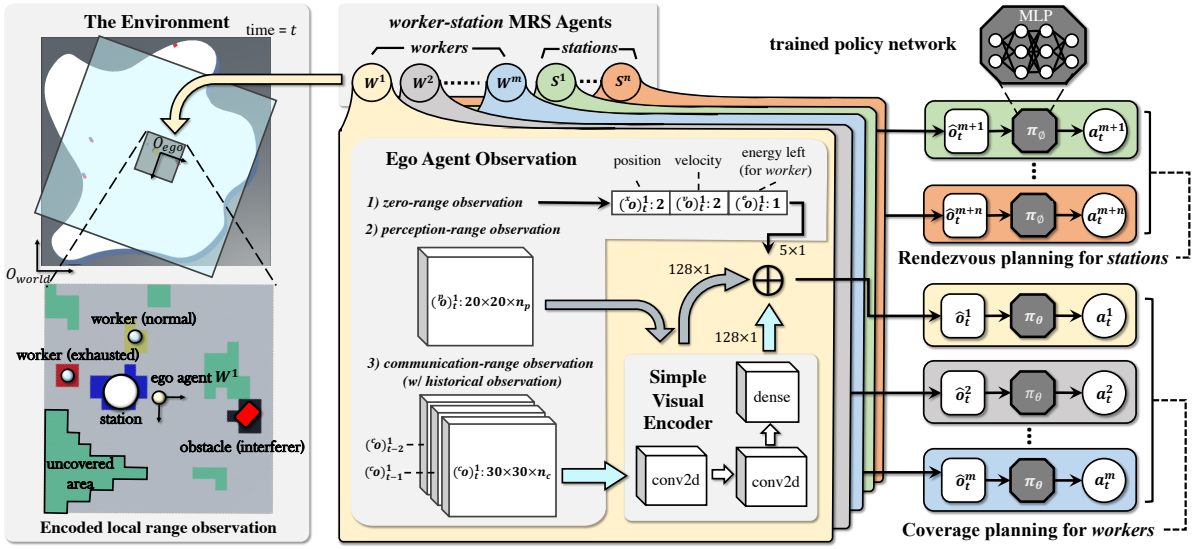



Fig. 4: Our DRL-based mCPP pipeline for the *worker-station* MRS: during planning, each agent receives its own zero-range, perception-range, and communication-range observations, and outputs the best action at each time step according to its trained policy network.

first two terms since the last two are simply penalty constants as introduced previously. Recall that the first component $({}^{c}r)^i$ is the covering reward for $i$-th *worker* at each time $t$, where positive rewards are given when a new area is covered. Once the coverage work is completed, the training episode will terminate with a completion reward $r_{\text{finish}}$:

$$({}^{c}r)^i = \begin{cases} r_{\text{finish}}, & \Omega = \bigcup_{t'=0}^{t} \bigcup_{i=1}^{k} \mathcal{C}_{t'}^{i} \\ r_{\text{cover}} \times \left( |\mathcal{C}_t^i| - |\mathcal{C}_{t-1}^i| \right), & \text{otherwise} \end{cases} \tag{5}$$

The second component $({}^{e}r)^i$ is a soft approximation modeling on the hard constraint of *worker*'s capacity. For *worker* $W^i$, it allows $p_t^i$ to be less than zero during training, which let $W^i$ still be able to move when $p_t^i \leq 0$ (i.e., no energy left). More specifically, such a soft approximation uses a truncated exponential function for $({}^{e}r)^i$ as below:

$$({}^{e}r)^i = \begin{cases} -1 \times \min\{1, \exp\left(p_t^i - p^e\right)\}, & p_t^i < p^e \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

where $p^e$ is the threshold indicating whether the *worker* is exhausted. Such design results from practical considerations: 1) direct modeling such hard constraint during training makes *worker* struggles to learn the "*cover and replenish*" behavior when energy is exhausted; 2) the truncated exponential penalty approximation with a relatively large derivative around $p^e$ makes *worker* aware of its exhausted status when $p_t^i$ approaches $p^e$. Note that the energy capacity hard constraint of *worker* is only modeled as a soft constraint during training; it is still a hard constraint (i.e., *workers* cannot move once $p_t^i \leq 0$) during planning.

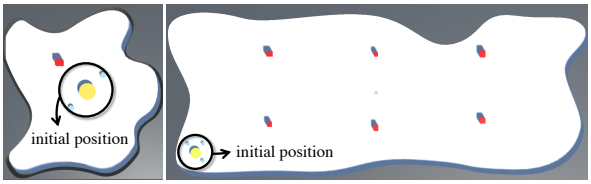
## V. Experiments & Results

### A. Implementation Details

Since we mainly focus on strategy-level planning problems in this paper, we use Unity and ML-Agents toolkit [30] to build the environment and system. The dynamic interferers are modeled in a loop to first move in a constant speed and a random direction within a given period, and then rotate with a random angle. Such a loop for interferers repeats until the coverage task finishes. Also, *worker* can only be replenished when it is exhausted (i.e., $p_t^i < p^e$) and near the *station*.

### B. Simulation Results

We modeled three simulation scenes in Unity to conduct simulation experiments, including ablation study and the coverage task performance comparison. As in Fig. 5-(a) and (b), two irregularly shaped scenes are used in simulation experiments, where robots of the *worker-station* MRS are initialized in the initial position. In addition, we modeled the CUHKSZ campus for coverage work in Fig. 5-(c), where




(a) The *star* scene (b) *The corridor scene*


(c) The CUHKSZ scene (for both test-cases *cuhksz-1* and *cuhksz-2*)

Fig. 5: Modeled simulation scenes in Unity. The target coverage areas are bounded within the grey obstacle areas.

| **test-case name** | ***star*** | ***corridor*** | ***cuhksz-1*** | ***cuhksz-2*** |
|---|---|---|---|---|
| **target area size** | 30×30 | 120×50 | 180×60 | 180×60 |
| ***worker* cover radius** | 4 | 4 | 2 | 2 |
| **# of *workers*** | 2 | 3 | 3 | 6 |
| **# of *stations*** | 1 | 1 | 1 | 2 |
| **# of *interferers*** | 1 | 6 | 6 | 6 |

TABLE II: Design details of simulation test-cases.

the buildings are considered static obstacles. Fig. 6 shows the motion trajectories of the *worker-station* using our planning method in simulation test-cases. Based on the three modeled

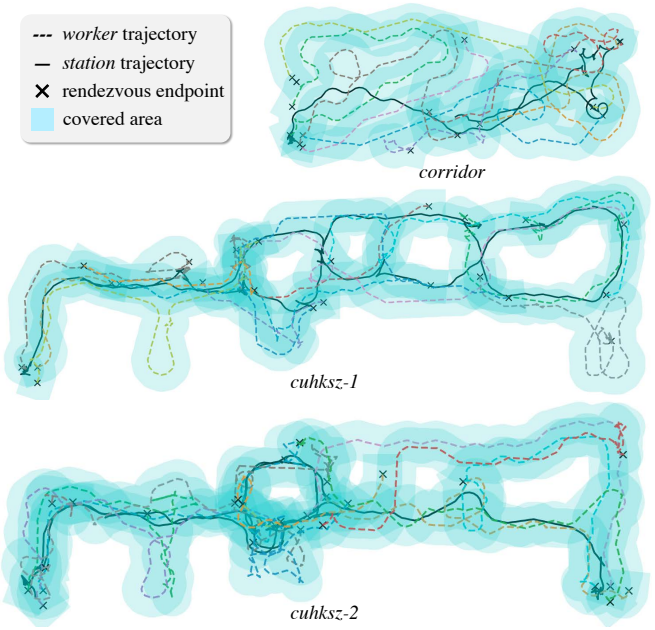
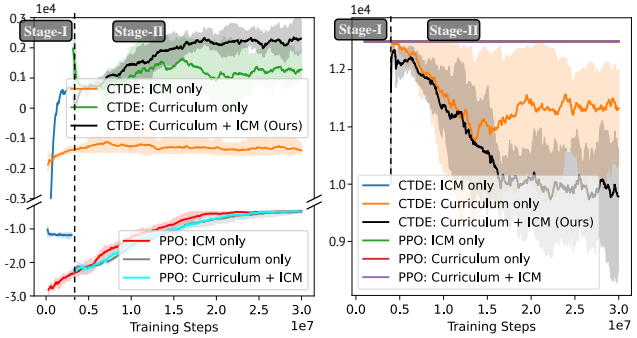



Fig. 6: motion trajectories of *worker-station* MRS using our method.

scenes, we designed four test-cases described in Tab. II. Note that in *cuhksz-2* test-case, only the left-bottom group of robots is included for the coverage work.

*1) Ablation Study:* to validate the effects on training performance brought by our curriculum learning design and ICM, we conducted ablation study in the *corridor* test-case. We also trained a centralized policy with PPO [31] to validate the benefits of the CTDE decentralization paradigm. In short, the PPO agent takes the observation of object-positions encoded images, which should cover the whole coverage area with high resolution as in the perception-range observation. Therefore, it is much larger than the image observations for each ego agent in CTDE. With the same visual encoder in Fig. 4, the feature vectors are fed into MLP of the same size to output the joint actions that are distributed to each robot. It is evident in Fig. 7 that a centralized policy using PPO failed in our problem. Such a failure is largely due to: 1) the training difficulties on a much larger network (about $1.5e6$ parameters with centralized PPO, $0.1e6$ parameters for both *worker* and *station* policies with CTDE); 2) and the lack of cooperation between agents with centralized PPO.

We now compare the results within the CTDE paradigm. For two-stage curriculum learning, as shown in Fig. 7, when training from scratch without curriculum, agents in the *worker-station* MRS struggle at a locally optimal policy and cannot finish the coverage task. When initializing from Stage-I, it provides basic policy networks for both *worker* and *station*, which vastly improves the sample efficiency and guides the training procedure. As for ICM, we first initialize policy networks from pre-trained Stage-I curriculum learning. As shown in Fig. 7-(b), the task finish time $t_{\text{finish}}$ shows that agents trained with ICM are better than agents trained without it, which reflects the reward gap between two training curves (green and black) in Fig. 7-(a). Such reward gap results from the earlier finish of the coverage task, which eliminates more accumulating time penalty $r_{\text{time}}$.




(a) Accumulated reward (b) Coverage task finish time

Fig. 7: Ablation study of two-stage curriculum learning, Intrinsic Curiosity Module (ICM) and centralized PPO in *corridor* test-case.

*2) Decomposition-based and Graph-based Baselines:* to evaluate the coverage task performance, we modified graph-based and decomposition-based mCPP methods to several heuristic baseline methods on discretized state space. In addition, since these offline centralized mCPP baseline methods have no dynamic collision avoidance ability with interferers, we adopt a *wait-and-move* policy for all baseline methods.

**Mobile-BCD:** for decomposition-based baseline method, we follow the Boustrophedon Cellular Decomposition (BCD) algorithm [32] and adopts it as the so-called *mobile-BCD* for our problem. We briefly describe the procedure: 1) the map is initially decomposed into cells via BCD; 2) in each cell, back-and-forth trajectories on the uncovered area are generated and evenly distributed to *workers*; 3) the *stations* always move to the nearest exhausted *worker* to replenish it.

**Static-MSTC*:** for graph-based mCPP baseline methods, we first modify the state-of-the-art mCPP algorithm MSTC* [33] into a static *stations* version, namely *static-MSTC**. In order to account for the continuous energy capacity of *workers* in our problem setting, the critical modification in static-MSTC* is the constraint approximation from the node-based energy capacity constraint in the original MSTC* to the travel time-based constraint as in Eq. 1.

**Mobile-MSTC*:** based on the static-MSTC* method, we further mobilize the *stations* and design the *mobile-MSTC** baseline as follows: 1) the target area is first decomposed into sub-regions via *k-means clustering*; 2) depth-first-search is applied to plan for the *stations* loaded with *workers*, to travel to the center of next uncovered sub-region; 3) at each sub-region, the *workers* cover the area via the static-MSTC* baseline method. Note that the $k$ value in the *k-means clustering* algorithm is chosen according to the capacity $c^i$ to make partitions suitable for efficient planning.

*3) Coverage Task Performance:* we compared the coverage task finish time $T_{\text{finish}}$ among our method and the above baseline methods on all the test-cases in Tab. II. The smaller $T_{\text{finish}}$ is, the better the planning strategy for the mCPP problem is. Note that to adopt the mobile-MSTC* baseline method, we decompose the CUHKSZ map into two

equal-sized areas, and each area runs the mobile-MSTC* baseline separately to finish the coverage work.

**Test-cases *star*, *corridor*, *cuhksz-2*:** we first compare static-MSTC* with mobile-MSTC* and mobile-BCD. In test-cases *star* and *corridor*, the performance of mobile-MSTC* and mobile-BCD is nearly the same as static-MSTC*. In test-case *cuhksz*-2, the mobile-MSTC* and mobile-BCD manage to improve task performance by mobilizing the *station* and planning for *station* and *workers* separately. However, there remains vast space for coverage task performance improvement; the mobile-station and mobile-BCD baselines that separately plan for *stations* and *workers* with dynamic interferers still perform inefficiently.

We now compare our method with baseline methods. In general, the comparison results on the three test-cases show that our planning method can generate good coordination behaviors of coverage planning and rendezvous planning for *workers* and *station*, which leads to a better performance in metrics of $t_{\text{finish}}$ after around $0.5e7$ to $1.5e7$ training steps. Compared with the mobile-MSTC* baseline in test-cases *corridor* and *cuhksz-2* with larger target areas, our method unlocks more benefits by mobilizing the *station* and utilizing the mobility of each robot in the MRS. Interestingly, when an exhausted *worker* leaves the perception or communication range of *station*, the rendezvous for recharge is still possible, as *stations* would explore to search exhausted *workers*.

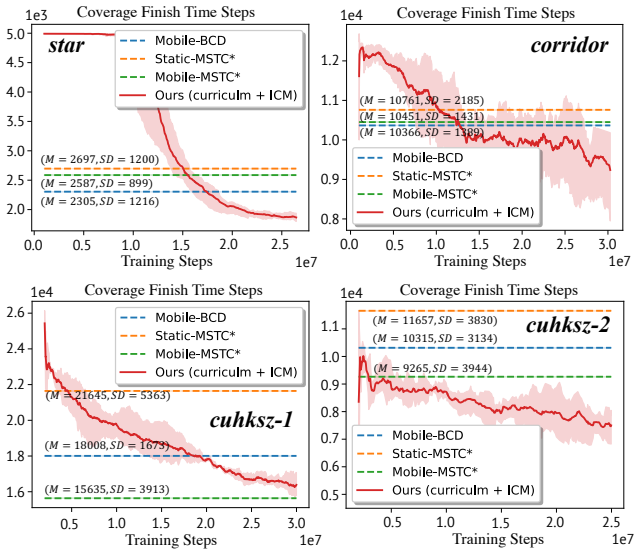
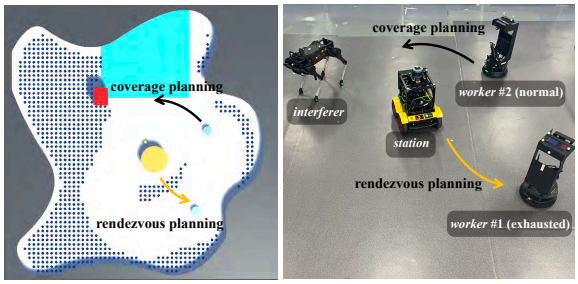



Fig. 8: Comparisons of the coverage task finish time.

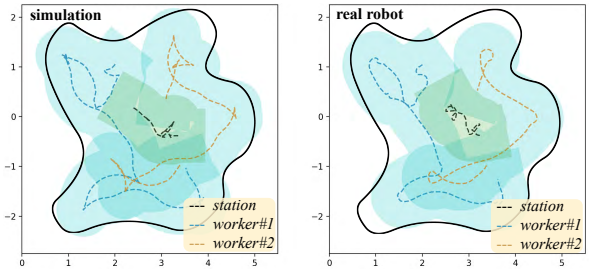
**Test-case *cuhksz-1*:** compared with test-case *cuhksz-2*, the MRS of test-case *cuhksz-1* works in the same CUHKSZ scene but with the numbers of *workers* and *stations* reduced in half. The performance of our method is nearly the same as mobile-MSTC* with the best performance, which is mainly due to the following reason: for less number of *workers* and a comparably smaller cover radius, *workers* with continuous action space would leave uncovered gaps during work, especially when trying to avoid static obstacles or dynamic interferers. These uncovered gaps require the *workers* to revisit some regions, making our method in this test-case perform not as well as in the other three test-cases.

### C. Real Robot Performance

As complementary to the simulated environments, we also conduct hardware experiments of our method on real robots. As shown in Fig. 9, we tested our method in *Star* scene, of which we made a replica in the real world. The *worker-station* MRS consists of two *workers* (black differential-driven wheeled robots) and one *station* (yellow skid-steer wheeled robot), the dynamic interferer is a quadruped robot. The *workers* are considered replenished once its distance to the *station* is smaller than a threshold. We use the PID controller for velocity commands of all robots, with a motion capture system providing their global position information.




(a) Real robot setup of the *worker-station* MRS




(b) Comparison of coverage area between simulation and real robot

Fig. 9: Real robot demonstration of our planning method.

Fig. 9-(a) depicts the *worker-station* MRS with our method, where the *station* is moving towards *worker* #2 to replenish it and *worker* #1 is executing coverage work. Fig. 9-(b) is a comparison of the coverage area between simulation and real robot, the blue area is covered by *workers* and the green area is the motion range of *station*, the whole coverage task in the real world took 140 seconds.

### D. Discussions

To the best of our knowledge, there is no existing online and simultaneous planning method for the *worker-station* MRS. Therefore, our choice of the baseline methods is naive heuristics-based planning approaches, which also need extensive research to reach optimal performance with fine-tuned hyperparameters. There are several limitations of our DRL-based planning method. First, when there is only a comparably small number of *workers* with a small cover radius, the unregulated trajectories of *workers* with continuous action space would leave more uncovered gaps needed for revisiting. Second, since our method mainly focuses on strategy-level planning for *workers* and *station* towards the coverage task, we use a relatively simple controller for the generated velocity actions to control real robots, which could cause a performance gap between simulation and reality.

## VI. Conclusions & Future Work

In this paper, we introduce the *worker-station* Multi-robot System (MRS) to solve the Multi-robot Coverage Path Planning (mCPP) problem, which can be generalized to various applications in the real world. We provide a fully cooperative multi-agent reinforcement learning formulation of the above problem, and propose an end-to-end decentralized online planning method based on Deep Reinforcement Learning. Our method simultaneously plans for *workers* and *station* to work together and utilize the mobility of each robot toward the coverage task. We conduct ablation study, simulation and real robot experiments and demonstrations. The experimental results show that our method is more efficient in planning for *workers* and *station*, and our method can better utilize the mobility of each robot compared with the mobile-station baseline method. For future work, there are two directions to further improve the coverage task performance based on our method. First, by regulating the trajectories with the graph-based mCPP method, there would be fewer missing gaps after *workers* covered an area. However, it might potentially raise the time for random dynamic collision avoidance. Second, by explicitly pre-allocating or negotiating which exhausted *workers* should the *stations* be responsible for, the *stations* can be more efficient to replenish specific *workers*.

## References


[1] Y. Liu and G. Nejat, "Multirobot cooperative learning for semiautonomous control in urban search and rescue applications," *Journal of Field Robotics*, vol. 33, no. 4, pp. 512–536, 2016.

[2] J. M. Palacios-Gasós, E. Montijano, C. Sagüés, and S. Llorente, "Distributed coverage estimation and control for multirobot persistent tasks," *IEEE transactions on Robotics*, vol. 32, no. 6, pp. 1444–1460, 2016.

[3] M. J. Schuster, M. G. Müller, S. G. Brunner, H. Lehner, P. Lehner, R. Sakagami, A. Dömel, L. Meyer, B. Vodermayer, R. Giubilato, *et al.*, "The arches space-analogue demonstration mission: towards heterogeneous teams of autonomous robots for collaborative scientific sampling in planetary exploration," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5315–5322, 2020.

[4] A. Couture-Beil and R. T. Vaughan, "Adaptive mobile charging stations for multi-robot systems," in *2009 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2009, pp. 1363–1368.

[5] Y. Litus, R. T. Vaughan, and P. Zebrowski, "The frugal feeding problem: Energy-efficient, multi-robot, multi-place rendezvous," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*. IEEE, 2007, pp. 27–32.

[6] R. Almadhoun, T. Taha, L. Seneviratne, and Y. Zweiri, "A survey on multi-robot coverage path planning for model reconstruction and mapping," *SN Applied Sciences*, vol. 1, no. 8, pp. 1–24, 2019.

[7] Y. Litus, P. Zebrowski, and R. T. Vaughan, "A distributed heuristic for energy-efficient multirobot multiplace rendezvous," *IEEE Transactions on Robotics*, vol. 25, no. 1, pp. 130–135, 2008.

[8] L. Wang, A. D. Ames, and M. Egerstedt, "Safety barrier certificates for collisions-free multirobot systems," *IEEE Transactions on Robotics*, vol. 33, no. 3, pp. 661–674, 2017.

[9] Xiaoming Zheng, Sonal Jain, S. Koenig, and D. Kempe, "Multi-robot forest coverage," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005, pp. 3852–3857.

[10] X. Zheng, S. Koenig, D. Kempe, and S. Jain, "Multirobot forest coverage for weighted and unweighted terrain," *IEEE Transactions on Robotics*, vol. 26, no. 6, pp. 1018–1031, 2010.

[11] A. C. Kapoutsis, S. A. Chatzichristofis, and E. B. Kosmatopoulos, "DARP: divide areas algorithm for optimal multi-robot coverage path planning," *Journal of Intelligent & Robotic Systems*, vol. 86, no. 3-4, pp. 663–680, 2017.

[12] I. Rekleitis, A. P. New, E. S. Rankin, and H. Choset, "Efficient boustrophedon multi-robot coverage: an algorithmic approach," *Annals of Mathematics and Artificial Intelligence*, vol. 52, no. 2, pp. 109–142, 2008.

[13] L. Collins, P. Ghassemi, E. T. Esfahani, D. Doermann, K. Dantu, and S. Chowdhury, "Scalable coverage path planning of multi-robot teams for monitoring non-convex areas," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 7393–7399.

[14] H. Azpúrua, G. M. Freitas, D. G. Macharet, and M. F. Campos, "Multi-robot coverage path planning using hexagonal segmentation for geophysical surveys," *Robotica*, vol. 36, no. 8, pp. 1144–1166, 2018.

[15] C. Sun, J. Tang, and X. Zhang, "FT-MSTC*: An efficient fault tolerance algorithm for multi-robot coverage path planning," in *2021 IEEE International Conference on Real-time Computing and Robotics (RCAR)*. IEEE, 2021, pp. 107–112.

[16] N. Mathew, S. L. Smith, and S. L. Waslander, "Multirobot rendezvous planning for recharging in persistent tasks," *IEEE Transactions on Robotics*, vol. 31, no. 1, pp. 128–142, 2015.

[17] K. Yu, A. K. Budhiraja, and P. Tokekar, "Algorithms for routing of unmanned aerial vehicles with mobile recharging stations," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 5720–5725.

[18] C. Sun, N. Kingry, and R. Dai, "A unified formulation and nonconvex optimization method for mixed-type decision-making of robotic systems," *IEEE Transactions on Robotics*, vol. 37, no. 3, pp. 831–846, 2020.

[19] S. Seyedi, Y. Yazicioğlu, and D. Aksaray, "Persistent surveillance with energy-constrained uavs and mobile charging stations," *IFAC-PapersOnLine*, vol. 52, no. 20, pp. 193–198, 2019.

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

[21] K. Zhang, Z. Yang, and T. Başar, "Multi-agent reinforcement learning: A selective overview of theories and algorithms," *Handbook of Reinforcement Learning and Control*, pp. 321–384, 2021.

[22] L. Busoniu, R. Babuska, and B. De Schutter, "A comprehensive survey of multiagent reinforcement learning," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 38, no. 2, pp. 156–172, 2008.

[23] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[24] T. Fan, P. Long, W. Liu, and J. Pan, "Distributed multi-robot collision avoidance via deep reinforcement learning for navigation in complex scenarios," *The International Journal of Robotics Research*, vol. 39, no. 7, pp. 856–892, 2020.

[25] R. Tallamraju, N. Saini, E. Bonetto, M. Pabst, Y. T. Liu, M. J. Black, and A. Ahmad, "Aircaprl: autonomous aerial human motion capture using deep reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6678–6685, 2020.

[26] A. Cohen, E. Teng, V.-P. Berges, R.-P. Dong, H. Henry, M. Mattar, A. Zook, and S. Ganguly, "On the use and misuse of absorbing states in multi-agent reinforcement learning," *arXiv preprint:2111.05992*, 2021.

[27] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 6382–6393.

[28] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, "Curiosity-driven exploration by self-supervised prediction," in *International conference on machine learning*. PMLR, 2017, pp. 2778–2787.

[29] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the 26th annual international conference on machine learning*, 2009, pp. 41–48.

[30] A. Juliani, V.-P. Berges, E. Teng, A. Cohen, J. Harper, C. Elion, C. Goy, Y. Gao, H. Henry, M. Mattar, and D. Lange, "Unity: A general platform for intelligent agents," 2020.

[31] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[32] H. Choset, "Coverage of known spaces: The boustrophedon cellular decomposition," *Autonomous Robots*, vol. 9, no. 3, pp. 247–253, 2000.

[33] J. Tang, C. Sun, and X. Zhang, "MSTC*: Multi-robot coverage path planning under physical constraints," in *2021 IEEE International Conference on Robotics and Automation*. IEEE, 2021.