Multi-Robot Exploration in Unknown Environments via Multi-Agent Deep Reinforcement Learning

Liyuan Deng[1], Wei Gong[1,2], Li Li[1,2]
[1] Department of Control Science and Engineering, Tongji University, Shanghai, China
[2] Shanghai Research Institute for Intelligent Autonomous Systems, Tongji University, Shanghai, China
{ dengliyuan99, weigong, lili }@tongji.edu.cn

2022 China Automation Congress (CAC) | 978-1-6654-6533-5/22/$31.00 ©2022 IEEE | DOI: 10.1109/CAC57257.2022.10055585

***Abstract*—Mobile robots have been widely used in hazardous environments to obtain information about surroundings for humans. The volume and efficiency of sensing data can be significantly increased if multiple mobile robots collaboration are exploited. In recent years, deep learning and reinforcement learning techniques have been applied to the field of robotics, which perform well in many tasks including exploration in unknown environments. In this paper, to address the multi-robot exploration problem, a multi-agent deep reinforcement learning (MADRL) based method with the centralized training and decentralized execution (CTDE) architecture was proposed. Extensive experimental results show that our method significantly improves the multi-robot exploration performance in unknown environments. On average, the proposed method uses 12.9% less travel distance and reduces 5.8% overlapping areas when compared with traditional methods.**

***Index Terms*—multi-robot system, deep reinforcement learning, unknown environment exploration**

I. Introduction

Exploration in unknown environments with mobile robots is one of the most important problems in the field of robotics. Mobile robots can be used to assist or replace humans to perform some tasks in hazardous or inaccessible environments, such as search and rescue (SAR) [1], [2], military exploration [3], and planet exploration [4].

In most cases of these scenarios, mobile robots have little or no prior knowledge of the environment. For example, the general shape of the area to be explored is known, but the structure inside that area is not. As a result, the robots need to build an accurate map of the environment when they perform the tasks.

According to the number of robot workers, the methods of the unknown environment exploration of robots can be divided into two categories [5]: single-robot and multi-robot methods. For single-robot methods, only one robot is used to perform the task. In this case, once the failure occurs the execution is terminated. Furthermore, it is difficult for the robot to balance the exploration of different subareas. Multi-robot methods could help to address these issues but it is more challenging to construct a multi-robot exploration system. Information exchange and cooperation patterns between robots need to be considered [6]. In real-world scenarios, it is difficult to guarantee the communication capability of the robot. Therefore, the decentralized robot exploration methods have been studied by [7], [8]. In this case, the robots communicate from time to time and make decisions locally without the centralized scheduling by a server.

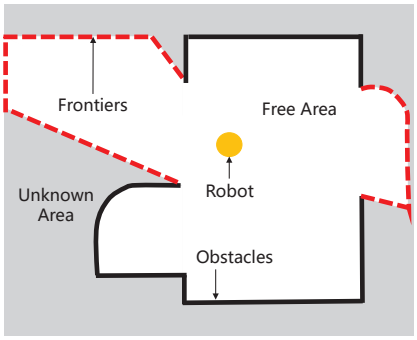

Fig. 1. A map of the environment created by a mobile robot during exploration. The boundaries (dashed red lines) between free space and unknown space are frontiers.

Simultaneous localization and mapping (SLAM) [9] is used for mapping in exploration tasks. The most common SLAM methods build a costmap of the environment from the sensor data, which represents the free space, obstacles, and unknown space on a grid map. The boundaries between free space and unknown space are called *frontiers*, as shown in Fig. 1. The map keeps being updated as the robot moves. For exploration tasks, the robot keeps moving to detect free areas and obstacles in the environment until there are no frontiers on the map, which means the completion of the task. The most popular environment exploration methods are based on frontiers [2], [10], which use a cost function or a utility function to decide a frontier as the target of the robot.

Deep learning and reinforcement learning techniques have been applied to robot exploration [2], [11], [12]. However, few studies are focusing on using the local observations of agents for exploration, which is a critical issue in real-world scenarios. To address it, we use Multi-Agent Deep Reinforcement Learning (MADRL) to solve the problem of multi-robot exploration.

This work has been supported in part by Shanghai Pujiang Program 21PJD076, in part by Shanghai Municipal Science and Technology, China Major Project under grant 2021SHZDZX0100, in part by Shanghai Research Institute of China Engineering Science and Technology Development Strategy, Strategic Research and Consulting Project, under grant 2022-DFZD-33-02, in part by Chinese Academy of Engineering, Strategic Research and Consulting Program, under grant 2022-XY-100, and in part by the National Natural Science Foundation of China under Grant 72171172.

In this paper, we develop a decentralized multi-robot exploration method which integrates MADRL with frontier exploration, which enables multiple robots to autonomously explore and build a map in unknown environments. Each agent makes decisions only from its local observations. To address the multi-robot exploration problem, we proposed a Multi-Agent Deep Deterministic Policy Gradient (MADDPG) [13] based method with the Centralized Training and Decentralized Execution (CTDE) architecture, which can handle the autonomous decision-making of multi-agents under partial observations.

The main contributions of this paper can be summarized as follows:

- A multi-agent deep reinforcement learning method for multi-robot exploration problem is proposed. This MADRL-based method facilitates the decentralized decision-making of target frontiers for the multi-robot system.
- Experiment results in the simulation environment show that our method has better performance than learning-based and traditional baselines.

The rest of the paper is organized as follows. Section II introduces the related works on robot exploration in unknown environments. Section III expounds the problem description and the proposed MADRL-based method. Section IV describes the training process of the networks and simulation results. Finally, Section V concludes the paper and presents the future work.

II. Related Works

In this section, we briefly discuss the recent studies on robot exploration in unknown environments. In recent years, learning-based methods are exploited to study the collaborative problems of multiple robots. Asynchronous Advantage Actor-Critic (A3C) algorithm is adopted in [2] to make the decision of the weights of two parts in the cost function. The robot then uses the weighted cost function to decide its target frontier. A multi-robot multi-target potential field (MMPF) exploration method is proposed in [6]. The repulsive potential field of obstacles and the gravitational potential field of frontiers are combined to guide robots to move. To implement a decentralized system, this work utilizes submaps [9] to fuse map frames of different robots. Moreover, the NetVLAD descriptor vector [14] is used for submap matching, thereby reducing the data interaction between robots. The Deep Deterministic Policy Gradient (DDPG) algorithm is used to generate continuous speed commands for robots to autonomously avoid obstacles while exploring [11]. In [12], a benchmark for robot exploration is developed. In this benchmark, the efficiency metrics of different methods of single/multi-robot exploration can be computed in different simulation environments. Graph Neural Networks (GNN) is adopted in [15], which enables robots to integrate the key information in the environment represented by a graph.

These works have made contributions to the field of robot exploration in unknown environments. Few of the existing studies directly use the local observations of agents for decision-making. Inspired by these works, the purpose of this paper is to develop multi-agent collaborative exploration method based on MADRL, which can help to solve the multi-robot collaborative exploration under the condition of partial observation.

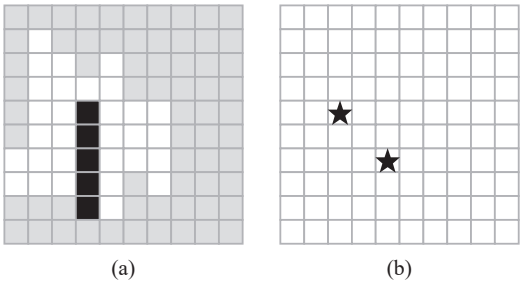

Fig. 2. The state space of the environment. The first part is the grid map of the current environment and the second part is the location of the agents.

III. Problem Description and MADRL-Based Method

In this section, we first described the problem of multi-robot exploration. Then we proposed our MADRL-based method to solve the problem. The details will be discussed in the following subsections.

A. Problem Description

Consider a 2D environment represented by the grid map $\mathcal{C}$ with the size $M \times N$. The entire environment is represented by the cells on the map and each cell corresponds to a small area. For each cell (m, n) in the grid map, it has three possible states. $c(m, n)$ is to describe the state of the cell, $c(m, n) = 0, 1, 2$ represents the cell is free, occupied or unknown, respectively. To make the exploration task achievable, we only consider maps that free space is fully connected.

We assume that robots are homogeneous and there are $\mathcal{U} \triangleq \{u = 1, 2, \cdots, U\}$ mobile robots. Each robot is equipped with sensors that can sense its surroundings within the sensing range l. Each robot starts from a free cell in the map randomly and decides a target frontier f_0^u by the decision-making method. The robot then moves towards the target while updating its local map $\mathcal{C}^u$. When the robot travels a distance, the frontiers are updated with the map. At time t, the location of robot u is (m_t^u, n_t^u), and its target frontier is f_t^u. During exploration, the robots will generate a trajectory set $\mathcal{P} \triangleq \{P^1, P^2, \cdots, P^U\}$. For a multi-robot exploration problem, the goal is to minimize the length sum of trajectories when completing the exploration task. And the completion of the task means that $\exists t_{\text{end}} > 0$, $\forall t \in [0, t_{\text{end}})$, $\bigcup_{u \in \mathcal{U}} \mathcal{C}_t^u \neq \mathcal{C}$, $\bigcup_{u \in \mathcal{U}} \mathcal{C}_{t_{\text{end}}}^u = \mathcal{C}$.

B. Reinforcement Learning Model

We model the problem as a Markov Decision Process (MDP). Coarsely speaking, an MDP involves the state-action-state pairs. The agent gets its observation o_t^u from environment s_t, then takes an action a_t^u. After that, the environment

transfers to a new state s_{t+1}, and the agent obtains a reward r_t^u.

The exploration process of the robot conforms to the Markov hypothesis, that is, the future state of the environment is only related to the current moment [5]. In this paper, we establish the following MDP to solve the considered problem.

State Space: The state space consists of two parts

$$\mathcal{S} \triangleq \{s_t = (s_{\text{map}}, s_{\text{loc}})\}. \tag{1}$$

The first part is the grid map of the current environment that has been explored. And the second part is the location of the agents, as shown in Fig. 2.

Observation Space: The observation space of an agent is its incomplete observation of the environment.

$$\mathcal{O} \triangleq \{o_t^u = (o_{\text{map}}^u, o_{\text{loc}}^u)\}, \forall u \in \mathcal{U}. \tag{2}$$

To simplify the problem, we assume that maps between agents are fused at every step. And the agent can obtain the location of itself but not others.

Action Space: $\mathcal{A} \triangleq \{a_t = (a_t^1, a_t^2, \cdots a_t^U)\}, a_t^u \in [-\pi, \pi]$. The agent will decide a direction to move according to its observations.

Reward: We set dense rewards for the agents, and it consists of the information gain g_t^u and penalty factor p_t^u.

$$r_t^u = \frac{g_t^u}{d_t^u} - p_t^u, \forall u \in \mathcal{U}, \tag{3}$$

where g_t^u is the information gain, defined by the difference in the size of the explored area, $g_t^u = \text{size}(\mathcal{C}_{t+1}^u) - \text{size}(\mathcal{C}_t^u)$. And d_t^u is the travel distance. Agents are penalized when explored areas overlap.

Discount Factor: Let $\gamma \in [0, 1]$ represent the discount factor for future rewards.

C. MADRL-Based Architecture

We adopted MADDPG [13] as the MADRL algorithm. In this subsection, the architecture of MADDPG is introduced.

1) Centralized Training and Decentralized Execution: MADDPG is an extension of Deep Deterministic Policy Gradient (DDPG) for multi-agent tasks, which is based on the classic actor-critic framework. Every agent has two deep neural networks actor $\pi^u(o_t^u)$ and critic $Q^u(s^j, a_j^1, \cdots, a_j^U)$, as shown in Fig. 3. Actor is the policy network that outputs a deterministic action from the agent's local observations and it does not depend on the observations of other agents. The actor network works during both the training process and the execution process. Critic is the value network and it outputs the value of the action to guide the actor network to update its parameters. Especially, it takes observations from all agents as input. The critic network works only during the training process.

2) Training Process: Algorithm 1 shows the details of the training process of MADDPG. Fig. 4 shows the relationship of each module of MADDPG algorithm for one robot u when training. At the very beginning, initialize the value network and policy network Q^u, π^u on each robot and copy them as

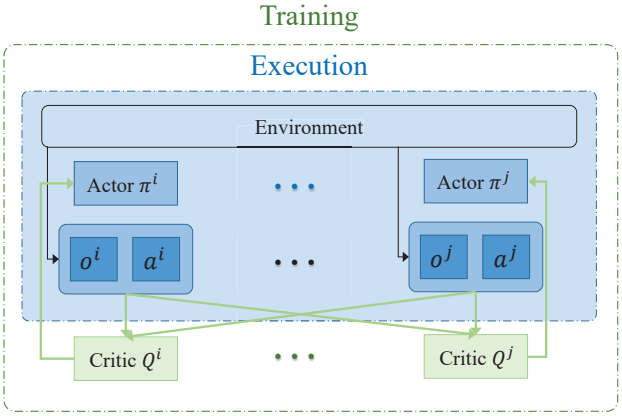

Fig. 3. The network structure of MADDPG.

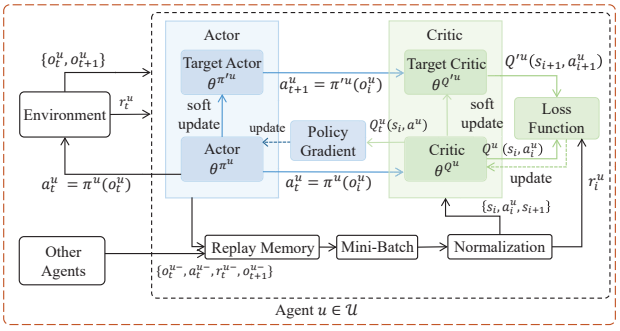

Fig. 4. The relationship of each module of MADDPG algorithm for one agent u when training.

its target networks Q'^u, μ'^u. Next, initialize the experience replay buffer B with the size $|B|$, which will store the transitions (s_t, a_t, r_t, s_{t+1}) during training. Then the networks are trained for *ep_max* episodes. In each episode, initialize the environment and the robots first. Each robot makes decisions from its local observations o_t^u of the environment s_t through π^u, and then gets a reward r_t^u. After that, the environment transfers to a new state s_{t+1}. The robot will store this process as a transition in buffer B. When the number of transitions is large than the batch size, the robot will take a batch of experiences to update the parameters of its networks. For the critic network, its parameters are updated by minimizing the loss function, as shown in Eqn. (4). As for the actor network, its parameters are updated by policy gradient, as shown in Eqn. (5)

$$L(\theta^{Q^u}) = \frac{1}{H} \sum_j (y_j^u - Q^u(s^j, a_j^1, \cdots, a_j^U))^2. \tag{4}$$

$$\nabla_{\theta^{\pi^u}} J \approx \frac{1}{H} \sum_j \nabla_{\theta^{\pi^u}} \pi_u(o_j^u) \nabla_{a_u} Q^u(s^j, a_j^1, \cdots, a_j^U). \tag{5}$$

Then the target networks are soft-updated by weight τ.

3) Executing Process: When executing MADDPG, for each robot, it only needs the actor network to produce an action a_t^u from its observation o_t^u. After robots complete their actions,

Algorithm 1: Training Process of MADDPG for Multi-Robot Exploration.

```
for Robot u ∈ 𝒰 do
    Initialize networks Q^u, π^u with weights θ^{Q^u}, θ^{π^u};
    Initialize target networks Q'^u, μ'^u with weights
      θ^{Q'^u}, θ^{π'^u};
end
Initialize replay buffer B with the size |B|;
for episode = 1:ep_max do
    Reset environment, obtain s_0;
    Set robots at the initial position in free space;
    Robots receive initial observations o_0^u;
    for t = 1:step_max do
        for Robot u ∈ 𝒰 do
            Decide an action a_t^u = π^u() and add a
              noise;
            Choose a goal frontier according to a_t^u;
            Plan a path to the goal by A* algorithm;
            Move along the path;
        end
        Update s_{t+1};
        for Robot u ∈ 𝒰 do
            Update o_{t+1}^u, r_t^u;
        end
        if Over 98% of the environment is explored
          then
            break;
        end
        Store transition (s_t, a_t, r_t, s_{t+1}) in B;
        for Robot u ∈ 𝒰 do
            Sample a batch of experiences from B;
            Update θ^{Q^u} by minimizing the loss;
            Update θ^{π^u} by policy gradient;
        end
        for Robot u ∈ 𝒰 do
            Soft update θ^{Q'^u} = τθ^{Q^u} + (1 − τ)θ^{Q'^u};
            Soft update θ^{π'^u} = τθ^{π^u} + (1 − τ)θ^{π'^u};
        end
    end
end
```

the environment transfers to s_t. Then each robot obtains its new observation o_{t+1}^u. This process is repeated until the exploration task is completed.

IV. Experiments

In this section, the experiment results are presented. We first present our experiment environment and then explain metrics for evaluation. Last, we compare our method with baselines and discuss the experiment results.

A. Experimental Settings

We modified the grid simulator [12] as the experiment platform. And all experiments are performed in a Ubuntu 18.04.6 LTS laptop with an Intel(R) Core(TM) i7-12700H CPU and an NVIDIA GeForce RTX 3050 Ti Laptop GPU.

We compared our method with three different baselines: cost function, greedy and DDPG.

- Cost function: The robot will choose the frontier with the least cost as its target.
- Greedy: The robot will choose the nearest frontier as its target.
- DDPG: DDPG is used for decision-making on each robot, with no information exchange between robots.

B. Metrics

We use two metrics to measure the performance of different methods.

- Sum of travel distances $D = \sum_{u \in \mathcal{U}} P^u$. This metric is to measure the efficiency of the exploration methods, which is computed by adding up the path lengths of each agent during an episode of exploration. We use $D_{90\%}$ and $D_{98\%}$ to represent the travel distances when 90% and 98% of the environment are explored.
- Overlap rate S_{o}. This is used to measure the collaboration effectiveness of the agents. It is not necessary to explore the same area with different agents. S_o is computed by

$$S_{\text{o}} = \frac{A_{\text{o}}}{A_{\text{total}}} = \frac{\sum_{u \in \mathcal{U}} A^u - A_{\text{total}}}{A_{\text{total}}}. \tag{6}$$

C. Neural Network Training

For hyperparameters of the network, we set learning rate of actor and critic that *lr_actor* $=$ 5e-5, *lr_critic* $=$ 5e-5, and *noise_rate* $= 0.1$ for sampling from a standard normal distribution, $\tau = 0.6$ for updating the target networks. The size of experiences buffer $|B| = 3000$ and the batch size for updating the value network is 50. We trained the networks for 1000 episodes and evaluated the networks every 10 episodes.

The reward curve in training process of MADDPG is shown in Fig. 5. At the beginning of training, both agents obtain low rewards and cannot finish the exploration. As the episode increases, the rewards obtained by agents gradually increase, which reflects the increase in exploration efficiency.

D. Experiment Results

We tested our method in an large-scale environment with the size 250×250 and compared the performance with three baselines: greedy, cost function, and DDPG. Two robots start from the starting points $(60, 60)$ and $(150, 150)$ in grid map respectively in each experiment. One mapping result of the collaborative exploration of two robots is shown in Fig. 6. The average performance of different methods in this environment is represented in Table I.

The experiment results show that the MADDPG-based method has better performance than other methods on the two metrics. In particular, the DDPG method has the worst performance in this scenario. There are great repetitive exploration areas between robots. The reason is that DDPG only uses local observations of the robot in both the training and execution process, while MADDPG uses global observations

Fig. 5. The reward curve of the training process of MADDPG.

TABLE I
THE AVERAGE PERFORMANCE OF DIFFERENT METHODS.

Method	$D_{90\%}$	$D_{98\%}$	S_o
MADDPG	404.4	525.4	12.9%
Cost Function	464.2	568.4	18.7%
Greedy	673.2	865.2	46.3%
DDPG	852.4	1013.0	68.9%

in the training process. Therefore, it can be seen that the CTDE feature of MADDPG makes great contributions to improving the performance of exploration.

V. CONCLUSION

In this paper, we have developed an exploration method that combines MADRL with frontier exploration for multiple robots to detect unknown environments collaboratively. Simulation experiments show that the proposed method has better performance than learning-based and traditional common baselines. Our future work is to develop a multi-robot collaborative exploration method applicable to more scenarios and test the method in physical simulation environments and physical robots.

REFERENCES

[1] S. Wirth and J. Pellenz, "Exploration transform: a stable exploring algorithm for robots in rescue environments," in *Proc. Int. Workshop Saf., Secur. Rescue Robot.*, Rome, Italy, 2007, pp. 1-5.

[2] F. Niroui, K. Zhang, Z. Kashino and G. Nejat, "Deep reinforcement learning robot for search and rescue applications: exploration in unknown cluttered environments," *IEEE Robot. Autom. Lett.*, vol. 4, no. 2, pp. 610-617, Apr. 2019.

[3] C. Kerr, R. Jaradat and N. U. Ibne Hossain, "Battlefield mapping by an unmanned aerial vehicle swarm: applied systems engineering processes and architectural considerations from system of systems," *IEEE Access*, vol. 8, pp. 20892-20903, Jan. 2020.

[4] Y. Huang, S. Wu, Z. Mu, X. Long, S. Chu and G. Zhao, "A multi-agent reinforcement learning method for swarm robots in space collaborative exploration," in *Proc. Int. Conf. Control, Autom. Robot.*, Singapore, 2020, pp. 139-144.

[5] I. Lluvia, E. Lazkano, and A. Ansuategi, "Active mapping and robot exploration: a survey," *Sensors*, vol. 21, no. 7, p. 2445, Apr. 2021.

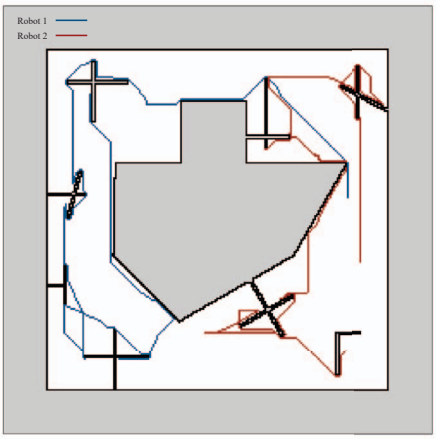

Fig. 6. One mapping result of the collaborative exploration of two robots. The trajectories of the two robots are represented in blue and red.

[6] J. Yu *et al.*, "Smmr-explore: submap-based multi-robot exploration system with multi-robot multi-target potential field exploration method," in *Proc. IEEE Int. Conf. Robot. and Autom.*, Xi'an, China, 2021, pp. 8779-8785.

[7] M. Geng, X. Zhou, B. Ding, H. Wang, and L. Zhang, "Learning to cooperate in decentralized multi-robot exploration of dynamic environments," in *Proc. Int. Conf. Neural Inf. Process.*, Siem Reap, Cambodia, 2018, pp. 40–51.

[8] M. Geng, K. Xu, X. Zhou, B. Ding, H. Wang, and L. Zhang, "Learning to cooperate via an attention-based communication neural network in decentralized multi-robot exploration," *Entropy*, vol. 21, no. 3, p. 294, Mar. 2019.

[9] W. Hess, D. Kohler, H. Rapp and D. Andor, "Real-time loop closure in 2d lidar slam," in *Proc. IEEE Int. Conf. Robot. Autom.*, Stockholm, Sweden, 2016, pp. 1271-1278.

[10] H. Umari and S. Mukhopadhyay, "Autonomous robotic exploration based on multiple rapidly-exploring randomized trees," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Vancouver, BC, Canada, 2017, pp. 1396-1402.

[11] J. Hu, H. Niu, J. Carrasco, B. Lennox and F. Arvin, "Voronoi-based multi-robot autonomous exploration in unknown environments via deep reinforcement learning," *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 14413-14423, Dec. 2020.

[12] Y. Xu *et al.*, "Explore-bench: data sets, metrics and evaluations for frontier-based and deep-reinforcement-learning-based autonomous exploration," in *Proc. IEEE Int. Conf. Robot. Autom.*, Philadelphia, PA, USA, 2022, pp. 6225-6231.

[13] R. Lowe, Y. Wu, A. Tamar, J. Harb, O. P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Proc. Adv. Neural Inf. Process. Syst.*, Long Beach, CA, USA, 2017, pp. 6379–6390.

[14] R. Arandjelović, P. Gronat, A. Torii, T. Pajdla and J. Sivic, "Netvlad: cnn architecture for weakly supervised place recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 6, pp. 1437-1451, Jun. 2018.

[15] H. Zhang, J. Cheng, L. Zhang, Y. Li and W. Zhang, "H2gnn: hierarchical-hops graph neural networks for multi-robot exploration in unknown environments," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 3435-3442, Apr. 2022.