IEEE Robotics and Automation Letters (RAL) paper presented at
2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

HMA-SAR: Multi-Agent Search and Rescue for Unknown Located Dynamic Targets in Completely Unknown Environments

Xiao Cao, Mingyang Li, Yuting Tao, and Peng Lu, *Member, IEEE*

***Abstract*—Multi-Agent Search and Rescue (MASAR) tasks, challenged by unknown environments and the unpredictable movements of unknown dynamic targets, suffer from inefficiencies in traditional map coverage techniques which require repeated sweeps. Addressing this, our study introduces a novel MASAR framework based on Multi-Agent Reinforcement Learning (MARL), featuring innovative elements like state, reward, and network structure design, alongside a Heterogeneous Curriculum Training algorithm and a hybrid decision mechanism. These components collectively enhance performance in dynamic environments, improve model generalization, and mitigate issues like sparse rewards and policy bias. In grid map simulations, our approach, HMA-SAR (Heterogeneous Multi-Agent Search and Rescue Framework), demonstrated consistent superiority over the traditional frontier-based method and other MARL algorithms, in metrics such as success rate, steps count, and the number of targets fetched. The practical applicability of our approach was further validated through experiments in Gazebo and real-world scenarios. Additionally, scalability tests in grid maps revealed substantial improvements in success rates and task completion times with increased agent deployment.**

***Index Terms*—Cooperative multi-agent search and rescue (MASAR), dynamic targets, heterogeneous curriculum training algorithm, hybrid decision mechanism, multi-agent reinforcement learning.**

I. Introduction

MULTI-AGENT Search and Rescue (MASAR) has become a focal point in research, notably for its applications in disaster scenarios such as fires and earthquakes [1], [2]. Traditional MASAR tasks involve agents covering the entire unknown terrain while simultaneously avoiding collisions, and searching for static targets [2], [3]. In contrast, this letter introduces a more complex task: agents are tasked with searching for unpredictable moving targets in unknown indoor areas. These agents are expected to go beyond basic exploration; they need to estimate target positions using a map that is only partially explored, along with time-sequenced trajectories, all while fostering effective collaboration. The scenario parameters, which include an area size that is variable but has an upper limit, and a constant initial target count, serve to clarify and define the context.

Traditional MASAR methodologies, primarily aimed at rapid and exhaustive environmental exploration [4], [5], often fall short in scenarios with unpredictably moving targets that may revisit previously explored areas. This results in missed targets only being identified by multiple coverage attempts, reducing overall efficiency. The dynamic nature of these targets presents agents with an exploration-exploitation dilemma: whether to venture into new areas or revisit old ones based on recent observations. To address this, most approaches employ mathematical programming methods, utilizing optimization solvers like CPlex [6] and branch-and-bound algorithms [7], or decentralized solutions that exploit specific problem features [8]. However, it's crucial to recognize that almost all existing MASAR planning methods depend on pre-known target motion dynamics and initial position data [9], which might not always be available in real-world settings. Conversely, learning-based methods offer advantages in such scenarios by eliminating the need for pre-evaluated target motion dynamics.

Multi-Agent Reinforcement Learning has emerged as an effective solution to the complex challenge of balancing exploration and exploitation. Existing studies, such as [10], [11], and [12], have shown progress in target search using agents' observations and joint maps, primarily focusing on static targets. Despite advances in handling non-adversarial, dynamic targets in known environments [13], there remains a significant research gap in addressing unknown environments with movable targets. Additionally, few studies have explored the challenges associated with sparse rewards in indoor settings, where agents struggle with long-distance navigation to unfamiliar areas, hindered by conventional reward structures that don't incentivize exploration.

To overcome limitations in dynamic target search on varied maps and terrains, we present the **HMA-SAR** (**H**eterogeneous **M**ulti-**A**gent **S**earch **a**nd **R**escue) Framework. This integrates a specialized state and reward design, a heterogeneous curriculum training paradigm, and a hybrid decision. Our contributions are:

- Proposing a novel multi-agent search and rescue framework based on multi-agent reinforcement learning that

Manuscript received 30 December 2023; accepted 24 April 2024. Date of publication 1 May 2024; date of current version 7 May 2024. This letter was recommended for publication by Associate Editor Rudolf Lioutikov and Editor Jens Kober upon evaluation of the reviewers' comments. This work was supported in part by General Research Fund under Grant 17204222 and in part by the Seed Fund for Collaborative Research and General Funding Scheme-HKU-TCL Joint Research Center for Artificial Intelligence. *(Xiao Cao and Mingyang Li are contributed equally to this work.) (Corresponding author: Peng Lu.)*

The authors are with the Adaptive Robotic Controls Lab (ArcLab), Department of Mechanical Engineering, The University of Hong Kong, Hong Kong, SAR 999077, China (e-mail: u3009678@connect.hku.hk; myli0823@connect.hku.hk; taoyt@connect.hku.hk; lupeng@hku.hk).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2024.3396097, provided by the authors.

Digital Object Identifier 10.1109/LRA.2024.3396097

addresses the limitations of existing map coverage methods in dynamic target search scenarios.
- Addressing sparse rewards in indoor environments through reward shaping based on the hybrid decision mechanism.
- Demonstrating the effectiveness of specific modifications in our training method through ablation studies and validating the superiority of our approach over the state-of-the-art MARL and traditional methods. Furthermore, the feasibility of our method is affirmed through both Gazebo simulations and real-world experiments.

The following sections will delve into related work (Section II), background and problem formulation (Section III), details of our framework (Section IV), and an analysis of the training process, results, and conclusions (Sections V, VI, and VII respectively).

II. Related Work

A. Conventional MASAR

To effectively tackle the autonomous exploration challenge in MASAR environments, rescue agents must traverse various areas within an unknown terrain to chart the environment and identify potential victims [14]. MASAR can be classified into two categories based on its objectives: target-oriented and coverage-oriented. In target-oriented MASAR, the distribution of the targets within the environment are initially known. Conversely, in scenarios where the MASAR environment remains unknown, coverage-oriented strategies are employed [15].

In the context of coverage-oriented MASAR, the primary goal for rescue agents is to navigate through unfamiliar terrains to map these areas and seek potential victims [4]. The most prevalent technique for exploring unknown spaces is frontier exploration [5]. This method propels the agents towards the frontiers, which are the boundaries between explored and unexplored territories. However, this approach can become extremely time-intensive when applied to extensive maps, as it requires navigating to frontier points across the entire map [16]. Other strategies for addressing coverage-oriented MASAR tasks include methods like the one in [17], which creates a spiral trajectory based on the agent's current location and areas already explored, to cover uncharted regions efficiently.

Despite these methodologies, most coverage-oriented solutions do not inherently integrate victim or target detection into their search algorithms. This omission restricts their adaptability and effectiveness in dynamic MASAR contexts [15]. Arnold et al. propose a cooperative MASAR system aimed at both victim detection and exploration in [18]. In this system, agents are guided by predefined sets of behaviors, limiting their flexibility in response to evolving situations.

B. DRL and Curriculum Learning-Based MASAR

Deep Reinforcement Learning (DRL) empowers agents with enhanced search and rescue capabilities through iterative training processes. In [19] and [14] an agent explores uncharted territories by selecting one point in the map as a navigation target. Transitioning to multi-agent systems, Ceyer's work in [10] marks a significant contribution by introducing novel observation encodings tailored for the independent actor-critic method, effectively addressing unique challenges. This approach employs MARL to coordinate agents in exploring unknown environments while minimizing collisions. Here, agents adopt a decentralized training and execution model, referred to as DTDE. In a similar vein, [11], [12] advocate for CTDE–centralized training for decentralized execution–and utilize MAPPO and MADDPG for terrain exploration. While the aforementioned methods are largely geared towards map coverage, [13] extend MARL techniques to target search applications. It outlines an approach for estimating the locations with the highest probability of containing targets, and then the agent is driven to arrive there.

Curriculum Learning, an approach guiding agents from simpler to more complex scenarios, has been shown to expedite convergence and has found applications in Search and Rescue (SAR) tasks. For instance, [20] introduces a strategy that incrementally increases the number of obstacles, while [21] gradually enlarges the environment's size. To mitigate the issue of catastrophic forgetting, where agents tend to forget previously learned skills as the curriculum becomes more challenging, [21] also puts forth a mixed training strategy. This involves interspersing simpler maps with complex ones, allowing agents to reinforce prior skills while assimilating new knowledge.

C. Sparse Reward

Sparse rewards arise from agents receiving rewards infrequently during their tasks. For instance, in exploration scenarios, agents are typically rewarded only when they discover unexplored regions. The study in [14] provides rewards solely at the completion of tasks or when unveiling new areas, whereas [19] implements negative rewards for agents until new regions are uncovered. However, these methods often overlook scenarios where agents have to traverse large areas that have already been explored and offer no rewards to guide agents' movements.

Addressing the challenge of sparse rewards often involves supplementing primary rewards with additional ones. One such method, as posited by [22], utilizes an Intrinsic Curiosity Module, motivating agents to explore further by assigning higher rewards to unfamiliar states. However, this approach necessitates the incorporation of three more neural networks, complicating the learning process. [14] modifies the action space where actions are not one-step movements but are paths leading to frontier points. The rewards, in this case, are tallied based on the number of grids newly discovered during this journey. Yet, when the entirety of the map has been scouted but some targets remain undetected, both this reward and action design falter.

III. Background and Fundamentals

A. Problem Formulation

The MASAR task, as we define it, integrates multi-agent exploration with target estimation, employing posterior data from multi-agent trajectories and exploration information. Each agent operates in a discrete action space, corresponding to the four cardinal directions: up, down, left, and right. The environments, characterized by walls, dead ends, and long corridors, are unknown indoor settings with no prior information on their size or layout, except for a known maximum size. The primary goals are efficient target search and collision avoidance. Agents are initiated from random yet strategically close positions, simulating real-world scenarios such as deploying rescue robots from

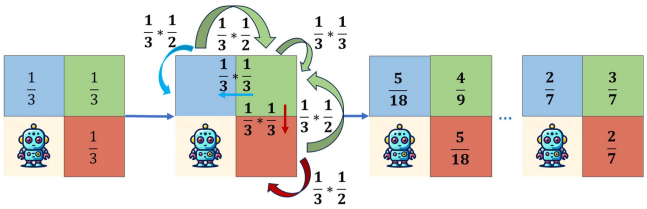

Fig. 1. The transition of the target's probability distribution over time is analyzed under the assumption that there is only one target and the observing robot remains stationary. Initially, the probability that the target is located in each square is set to $\frac{1}{3}$. The probability distribution converges to $\frac{2}{7}$, $\frac{3}{7}$, $\frac{2}{7}$.

adjacent areas. They are tasked with developing search strategies for targets, utilizing their trajectories and exploration data. *Based on the agents' trajectories and exploration information, and assuming random target movements, agents can infer the probability of targets being in various locations.*

To illustrate this, consider a simplified example shown in Fig. 1, where both the robot and target can move to neighbour grids or stay still. Initially, all squares have a target probability of $\frac{1}{3}$. When considering the next step, we assume the robot stays still and does not detect the target, starting our analysis with the blue square.

- *Loss*: If the target **moves out** of the blue square, there is a probability 'loss' of $\frac{1}{3} \times \frac{1}{2} = \frac{1}{6}$, since it can stay or move right.
- *Gain*: The blue square can 'gain' probability of $\frac{1}{3} \times \frac{1}{3} = \frac{1}{9}$ if the target **moves into** it from the green since the target in the green square can stay, move left or move down.

Combining these results, we derive the updated probability for the target in the blue square by subtracting the 'loss' and adding the 'gain', yielding $\frac{1}{3} - \frac{1}{6} + \frac{1}{9} = \frac{5}{18}$. The probability distribution attains stability when 'loss' equals 'gain' between adjacent squares. In the context of the blue and green squares, this balance results in a $2:3$ ratio when 'loss' equals 'gain'. Applying similar logic, we find that the stable ratio of probabilities for the blue, green, and red squares is $2:3:2$. This equilibrium reflects an equalized probability flux between these squares, implying that probability distribution is stable.

B. Challenges and Open Issues

The field of multi-agent reinforcement learning for exploration is fraught with challenges: **Environmental Dynamics**, **Model Generalization**, **Sparse Reward**, and **Policy Bias**, both inherent to MARL and specific to exploration tasks.

1) *Environmental Dynamics*: Environmental dynamics refers to the changing probabilities of state transitions and variations in rewards for actions within an environment. It comes from the uncertainties in the decisions made by other agents, various shapes, sizes, and elements in the environmental maps, and the stochastic distribution of target locations.
2) *Model Generalization*: Considering the diverse range of exploration maps, developing strategies with broad applicability is crucial.
3) *Sparse Reward*: Standard reward systems for exploration tasks do not incentivize agents to revisit previously mapped areas. This becomes more problematic in complex terrains, such as dead ends. For instance in Fig. 2, when an agent goes down a long dead end, the way back, already mapped, offers no new exploration rewards. This factor intensifies the challenge of efficient exploration.
4) *Policy Bias*: Agents can collide with obstacles frequently in the early stage of training, causing penalties frequently. This can lead agents to adopt conservation strategies that do not explore tight paths even though there may be a large area behind them.

Fig. 2. Black grid is an obstacle, white grids are explored, and the green grid represents an unknown grid.

To tackle these challenges, we propose a new training paradigm: combining a unique multi-agent reinforcement learning algorithm with a specialized curriculum training method. The following sections will detail this innovative approach.

IV. Heterogeneous Multi-Agent Search and Rescue Framework: Training Paradigm and Hybrid Decision Mechanism

We present a novel multi-agent search and rescue framework, termed the HMA-SAR as outlined in Fig 3. It consists of three key components.

1) State, Reward, and Network Structure Design: A specialized state and reward function design is tailored specifically for addressing the **Sparse Reward** encountered in search and rescue tasks, as described in Subsection III-B. This is coupled with a robust neural network structure explicitly designed to tackle **Model Generalization**.
2) Heterogeneous Curriculum Learning: A Heterogeneous Multi-Agent Learning Algorithm is formulated in tandem with a Curriculum Learning algorithm. It is specially designed to tackle **Policy Bias** and the **Environmental Dynamics** caused by other agents' decisions and the map differences in shapes, and sizes. However, we find it is robust enough to work well with the stochastic distribution of target locations. To address **Policy Bias**, we forbid collision in training to offer agents greater opportunities to explore narrow paths, which helps eliminate the biased performance that overly favours the exploration of open areas. Regarding **Environmental Dynamics**, we implement sequential policy updates to allow agents to effectively understand and predict each other's strategy. To mitigate the impact of map differences, we adopt a curriculum learning approach gradually increasing the effect of the difference in maps' sizes and shapes by restricting the maximum environment step t (the number of steps an agent can take before resetting the environment). This strategy is illustrated in Fig. 3 b). For two different maps depicted in i), limiting the maximum environment step t, as shown in ii), results in identical explorable areas, represented by circles of the same radius. As t increases (from ii) to iii) and then to iv)), the differences in explorable areas are more pronounced.
3) Hybrid Decision Mechanism: This serves as a conditional trigger for deciding whether to employ a rescue algorithm for action generation instead of the neural network during

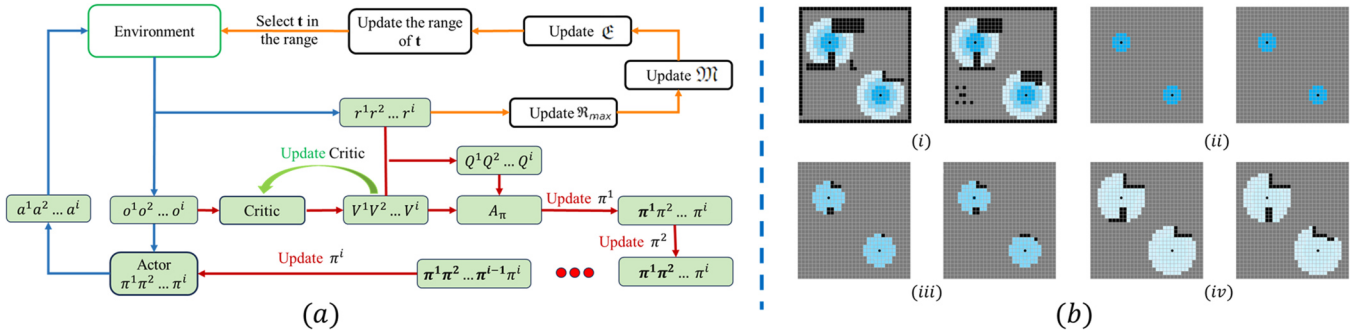

Fig. 3. The subfigure (a) outlines the framework of HMA-SAR, comprising three distinct parts: blue lines represent the interactions between agents and their environment, red lines illustrate the process of sequential policy updates, and orange lines demonstrate the curriculum learning process. Subfigure (b) shows how curriculum learning mitigates the impact of map differences (as seen in (i)) by capping the maximum environmental step. Through panels (ii), (iii), and (iv), it becomes apparent that as this maximum step limit increases, the diversity in explorable areas (blue grids) for agents on the same maps also grows. Black grids indicate obstacles and grey grids denote areas beyond exploration.

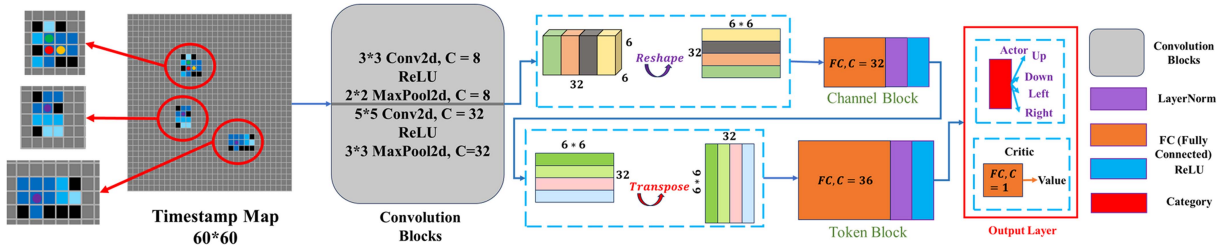

Fig. 4. Here is the architecture of our neural network MLP-Mixer. The input (timestamp map) includes four types of data: the unknown area data (grey grids), the occupied area data (black grids), the vacant area data (blue grids), and coordinate data (colourful dots). After convolution blocks, we add a reshape operation to flatten the data. After the channel block, the data is transposed. In the HMA-SAR context, as an actor, the network outputs four action probabilities; as a critic, it outputs a single state value. Notations "3*3"("2*2" or "5*5") and "C" specify kernel sizes and output channels, respectively.

tests. It aims to prevent agents from getting stuck in some maps and real-world scenarios.

A. State, Reward, and Network Structure Design

We define the state of each agent as a timestamp map as shown in Fig. 4, containing four types of data:

- Unknown area data: The grey grids of timestamp map in Fig. 4. Symbolized by 0.5, it represents areas that are yet to be explored and might contain obstacles or be empty.
- Occupied area data: The black grids of timestamp map in Fig. 4. Indicated by 1, it denotes the presence of obstacles in those areas.
- Vacant area data: Represented by the blue grids in the timestamp map shown in Fig. 4, with shades corresponding to values between 0 and 0.3. These values reflect the elapsed time since the last observation, with lower values indicating more recent sightings.
- Coordinate data: In the timestamp map for agent i, the trajectory of agent i over the last three environment steps is represented by dots in yellow, red, and green, corresponding to the values -4.5, -5.5, and -6.5 respectively. The positions of other agents are denoted by purple dots, each with a value of 5.5. Overlaps use the highest value.

Note that the assigned values, including those for coordinates, are chosen arbitrarily for model clarity as long as they are different from each other. The focus is on differentiation rather than the values themselves.

With such a state, the reward function is defined as $r = r_e + r_s$. r_e is defined as:

$$r_e = 0.1(4 * m_{ex} - 30 * i_c + i) \tag{1}$$

where m_{ex} represents the number of new open grids uncovered by an agent's action a_t from state s_t to s_{t+1}. The variable i_c is set to 1 if the agent collides with other agents or obstacles, and 0 otherwise. The term i, a repeat penalty, is applied if an agent's position repeats three times within the last 10 environmental steps. In this case, the A^* algorithm calculates the shortest path length p_t to the grid with the highest timestamp value at environmental step t, as an alternative to targeting an unknown grid or the most recently explored grid. The value of i is determined as 1 if $p_t < p_{t+1}$ at step t; otherwise, it is set to -1.

r_s is defined as

$$r_s = \begin{cases} 0.03 * \sum_{i=0}^{N_v}(1 - \frac{t_i}{0.3}) - 1, & \text{when not finding targets,} \\ 0.1, & \text{when finding a target.} \end{cases} \tag{2}$$

where N_v corresponds to the quantity of perceivable open grids within the current viewing range of an agent, t_i is the value of grid i in timestamp map between 0 and 0.3 as described in vacant area data.

To handle timestamp map inputs, we propose a model inspired by MLP-Mixer [23], serving as the structure for both the actor and critic as depicted in Fig. 3. As illustrated in Fig. 4, the neural network comprises convolution blocks, a channel block, a token block, and an output layer. The output layer is a category layer

when the network serves as the actor and a Fully Connected layer when the network serves as the critic. Not accounting for batch size dimensions, the data undergoes reshaping to $(32, 6 \times 6)$ after the convolution blocks and is transposed to $(6 \times 6, 32)$ following the channel block. Serving as an actor, the network uses the category layer to output probabilities for each possible action, with actions selected based on these probabilities using the roulette wheel selection method. Conversely, when configured as a critic, the network outputs a state value, computed by a Fully Connected layer, assessing the current state's worth for decision-making processes.

B. Heterogeneous Curriculum Training

The Heterogeneous Curriculum Training (HCT) method, derived from the Heterogeneous-Agent Proximal Policy Optimisation (HAPPO) [24], is depicted in Fig. 3. The blue lines present that agents interact with the environment. The orange lines present the curriculum learning. The red lines illustrate the process of sequential policy updates. Following the orange lines, in HCT, we forbid the collision first and set a constraint on the maximum environment step t, which is in the range of $[\mathfrak{E}_{min}, \mathfrak{E}_{max}]$. The t is updated following the orange lines in Fig. 3. At the beginning of training, t equals to $\mathfrak{E}_{min}$. In each epoch, HCT records the average reward $\mathfrak{R}$ and updates the maximum value $\mathfrak{R}_{\max}$. The consecutive training epochs when no improvement is observed in $\mathfrak{R}_{\max}$ is denoted as $\mathfrak{M}$. Then we calculate $\mathfrak{E}$

$$\mathfrak{E} = \min(\mathfrak{E} + \mathfrak{E}_{\min}, \mathfrak{E}_{\max}), \text{ when } \mathfrak{M} > 100 \tag{3}$$

The t is randomly selected in a range about $\mathfrak{E}$ and we implement the t in the advantage function

$$\begin{aligned} &A_{\pi}^{i_{1:m}}\left(s_t, \boldsymbol{a}_{\boldsymbol{t}}^{i_{1:m}}\right) = Q_{\boldsymbol{\pi}}^{i_{1:m}}\left(s_t, \boldsymbol{a}_t^{i_{1:m}}\right) - V_{\boldsymbol{\pi}}(s_t) \\ &\text{subject to } t \in \left(\mathfrak{E} - \frac{\mathfrak{E}_{\min}}{2}, \min\left(\mathfrak{E} + \frac{\mathfrak{E}_{\min}}{2}, \mathfrak{E}_{\max}\right)\right) \end{aligned} \tag{4}$$

For the sake of simplicity in expression, we will omit t and the related constraints in the subsequent formulas In (4), $A_{\pi}^{i_{1:m}}(s, \boldsymbol{a}^{i_{1:m}})$ defines the joint advantage value of agent $i:m$ executing action $\boldsymbol{a}^{i:m}$ given the state s. $Q_{\boldsymbol{\pi}}^{i_{1:m}}(s, \boldsymbol{a}^{i_{1:m}})$ defines the joint state-action value of agent $i:m$. The $Q_{\boldsymbol{\pi}}^{i_{1:m}}(s, \boldsymbol{a}^{i_{1:m}})$ is defined as

$$Q_{\pi}^{i_{1:m}}\left(s, \boldsymbol{a}^{i_{1:m}}\right) \triangleq \mathbb{E}_{\mathbf{a}^{-i_{1:m}} \sim \boldsymbol{\pi}^{-i_{1:m}}}\left[Q_{\boldsymbol{\pi}}\left(s, \boldsymbol{a}^{i_{1:m}}, \mathbf{a}^{-i_{1:m}}\right)\right] \tag{5}$$

where $\mathbf{a}^{-i_{1:m}}$ defines the actions taken by all the agents except agent $1 \sim m$. In (4), agents have to memorize other agents' actions to calculate joint Q value. To make each agent focus on their own actions, HCT introduces dynamic programming to express the joint advantage function as the summation of single advantage functions

$$\begin{aligned} A_{\pi}^{i_{1:m}}\left(s, \boldsymbol{a}^{i_{1:m}}\right) &= Q_{\boldsymbol{\pi}}^{i_{1:m}}\left(s, \boldsymbol{a}^{i_{1:m}}\right) - V_{\boldsymbol{\pi}}(s) \\ &= \sum_{k=1}^{m}\left[Q_{\boldsymbol{\pi}}^{i_{1:k}}\left(s, \boldsymbol{a}^{i_{1:k}}\right) - Q_{\boldsymbol{\pi}}^{i_{1:k-1}}\left(s, \boldsymbol{a}^{i_{1:k-1}}\right)\right] \\ &= \sum_{k=1}^{m} A_{\boldsymbol{\pi}}^{i_k}\left(s, \boldsymbol{a}^{i_{1:k-1}}, a^{i_k}\right). \end{aligned} \tag{6}$$

Following the blue lines in Fig. 3, HCT updates each agent's policy sequence by sequence, the objective of the agent m is defined as

$$\begin{aligned} &\mathbb{E}_{\mathbf{a}^{i_{1:m-1}} \sim \bar{\pi}^{i_{1:m-1}}, \mathbf{a}^{i_m} \sim \hat{\pi}^{i_m}}\left[A_{\pi}^{i_m}\left(s, \mathbf{a}^{i_{1:m-1}}, \mathrm{a}^{i_m}\right)\right] \\ &= \mathbb{E}_{\mathbf{a} \sim \pi}\left[\left(\frac{\hat{\pi}^{i_m}\left(\mathrm{a}^{i_m} \mid s\right)}{\pi^{i_m}\left(\mathbf{a}^{i_m} \mid s\right)} - 1\right)\right. \\ &\left.\times \frac{\bar{\pi}^{i_{1:m-1}}\left(\mathbf{a}^{i_{1:m-1}} \mid s\right)}{\pi^{i_{1:m-1}}\left(\mathbf{a}^{i_{1:m-1}} \mid s\right)} A_{\pi}(s, \mathbf{a})\right] \end{aligned} \tag{7}$$

where π is the old policy, $\bar{\pi}^{i_{1:m-1}}$ is the just updated joint policy of agent $1 \sim m$, $\hat{\pi}^{i_m}$ is the target policy of agent m. HCT optimizes the policy $\hat{\pi}^{i_m}$ parameter θ^{i_m} by maximising the objective function at time k:

$$\theta_k^{i_m} = \arg\max_{\theta_k^{i_m}} \mathbb{E}_{\substack{t=k, \\ \mathbf{a}^{i_{1:m-1}} \sim \bar{\pi}^{i_{1:m-1}}, \\ \mathbf{a}^{i_m} \sim \hat{\pi}^{i_m}}}\left[A_{\pi}^{i_m}\left(s, \mathbf{a}^{i_{1:m-1}}, \mathrm{a}^{i_m}\right)\right] \tag{8}$$

Use GAE [25] to estimate the advantage function and consider the clip mechanism in PPO, the final policy loss function is

$$\begin{aligned} L^{policy} = \mathbb{E}_{\mathbf{s} \sim \rho_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}, \mathbf{a} \sim \pi_{\boldsymbol{\theta}_{\boldsymbol{k}}}}\Bigg[&\min\left(\frac{\pi_{\theta^{i_m}}^{i_m}\left(\mathrm{a}^i \mid \mathrm{s}\right)}{\pi_{\theta_k^{i_m}}^{i_m}\left(\mathrm{a}^i \mid \mathrm{s}\right)} M^{i_{1:m}}(\tilde{\mathrm{s}}, \mathbf{a}),\right. \\ &\left.\left.\operatorname{clip}\left(\frac{\pi_{\theta^{i_m}}^{i_m}\left(\mathrm{a}^i \mid \mathrm{s}\right)}{\pi_{\theta_k^{i_m}}^{i_m}\left(\mathrm{a}^i \mid \mathrm{s}\right)}, 1 \pm \epsilon\right) M^{i_{1:m}}(\tilde{\mathrm{s}}, \mathbf{a})\right)\right] \end{aligned} \tag{9}$$

where $M^{i_{1:m}}$ is defined below

$$M^{i_{1:m}} = \frac{\overline{\boldsymbol{\pi}}^{i_{1:m-1}}\left(\mathbf{a}^{i_{1:m-1}} \mid s\right)}{\boldsymbol{\pi}^{i_{1:m-1}}\left(\mathbf{a}^{i_{1:m-1}} \mid s\right)} A_{\pi}(s, \mathbf{a}) \tag{10}$$

The value loss is

$$\begin{aligned} L^{value} = \max\Big(&H\left(R - V_{\boldsymbol{\pi}}(s), \delta\right), \\ &H\left(R - (V_{\text{old}}(s) + \text{clamp}(V_{\boldsymbol{\pi}}(s) - V_{\text{old}}(s), -c, c)), \delta\right)\Big) \end{aligned} \tag{11}$$

where R is the return of state s, and c is the clipping parameter that constrains the updated value estimates within a specified range around the original estimates. The function H is

$$H(x, \delta) = \begin{cases} \frac{1}{2}x^2 & \text{if } |x| \le \delta, \\ \delta\left(|x| - \frac{1}{2}\delta\right) & \text{otherwise.} \end{cases} \tag{12}$$

C. Hybrid Decision Mechanism

The hybrid decision mechanism, consisting of a neural network and a rescue algorithm, is used only in the testing phase. It is a conditional trigger. When an agent i's position repeats three times within the last 10 (an empirical number) environmental steps, the agent will generate actions using the rescue algorithm rather than using the neural network. The algorithm first identifies the coordinates of the nearest undiscovered grid g_i. If all grids have been explored, it selects the nearest grid with the largest timestamp value of vacant area data. It then calculates the path p_i using the A^* algorithm. Importantly, this mechanism dynamically recalculates both the grid g_i and the path p_i if another agent explores the designated grid g_i, thus adapting to the changing environment.

Fig. 5. Environment examples, each has a unique size, and the shape is regulated to (60,60) by padding.

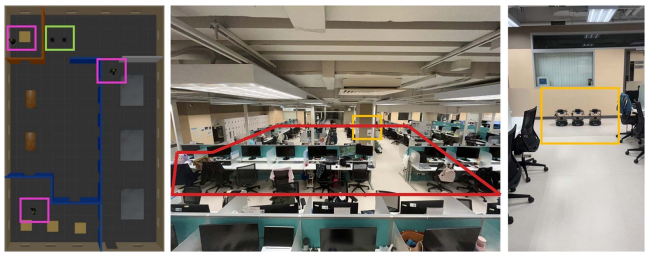

Fig. 6. Comparative view of the Gazebo simulation environment (left), real-world environment layout (middle), and TurtleBots (right). In the left image, targets are within a purple frame and agents are within a green frame. The middle image outlines the search and rescue area in red and shows the initial position of the TurtleBots in orange, with precious positions in the right image.

V. Training

The HMA-SAR Framework was trained across a dataset [19] comprising 5663 grid world environments, each sized 60×60. These environments showcase a variety of open area sizes, as depicted in Fig. 5. To simulate noise interference, the visual range of one agent fluctuates between 4 and 5 units.

The training adopted a curriculum learning approach, initially setting $\mathfrak{E}$ and the maximum environment step t as 10. If the average reward of a training epoch does not surpass the highest reward of the previous 200 consecutive epochs, $\mathfrak{E}$ is increased by 10 and t is randomly selected from the range $(\mathfrak{E}-5, \min(\mathfrak{E}+5, 250))$. The training concludes once $\mathfrak{E}$ exceeds 250. This process was executed on a system equipped with a 3080Ti, with a batch size of 1850. As shown in Fig. 5, grid maps are employed for training. Agents start randomly within a 3-unit radius circle, simulating close-proximity starts. They move one unit per step in four directions (up, down, left and right). Six targets, initiated randomly, also move one unit per step in four directions without colliding with obstacles, staying within the map limits. Two training scenarios depicted in Fig. 7 are introduced: In scenario α, dynamic targets are initiated at the beginning, whereas in scenario β, targets are initiated after 50% of the area has been explored, and at least two targets are initiated in the explored areas, representing missed detection scenarios where targets have been overlooked.

VI. Experiment Results

A. Experiment Setting

Experiments have been conducted in three settings: grid worlds (see Fig. 5), Gazebo, and real-world physical environments (see Fig. 6). We have demonstrated HMA-SAR's superior performance through quantitative analysis in the grid map and have validated its feasibility with TurtleBots in both Gazebo and real-world environments. For the grid maps, we have randomly chosen 250 scenarios from the test sets and have limited

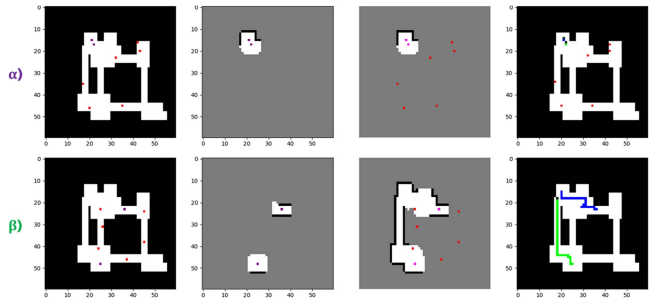

Fig. 7. *Scenarios* $\alpha\&\beta$: α presents the scenario where targets are initiated at the beginning and β presents the scenario where targets are initiated after 50% area has been explored. Targets are distributed in both explored and unexplored areas.

the maximum number of steps in each environment to 250. In these grid worlds, collisions have been prohibited; agents have remained stationary if moving would result in a collision. Additionally, agents have been positioned nearby. In Gazebo, we have initiated two TurtleBots and also introduced three walking men, who are designated as dynamic targets, as shown in Fig. 6. In the real world, three TurtleBots and walking men carrying colourful boards served as dynamic targets. There were a total of six targets, and the three walking men reset their positions once all were detected.

B. Comparative Study

We have conducted a comprehensive benchmark study to evaluate the efficacy of the HMA-SAR method in contrast with HAPPO, HAPPO-NC, MASAC [26], MAPPO [27], and the frontier-based exploration technique [28].

1) HAPPO: The state-of-the-art MARL algorithm, permits collision actions during training. The environment resets upon any collision.
2) HAPPO-NC: This is HAPPO but collisions during training are disallowed. The agent will stay still if its action leads to a collision.
3) MASAC: One of the common MARL algorithms, agents update policies synchronously considering return and the richness of actions.
4) MAPPO: One of the common MARL algorithms, providing stable synchronous policy updates.
5) Frontier-based exploration: Agents are programmed to gravitate toward the nearest frontier points.

We propose five indicators, 'Success Rate (SR)', representing the percentage of successful tests; 'Average Step Count (ASC)'–which denotes the average number of environment steps per test across 250 different maps; 'Variance of Step Count (VSC)', indicating the variability in the number of steps per test; 'Average of Fetched Targets(AFT)', reflecting the average number of targets detected; and 'Variance of Fetched Targets (VFT)', showing the variability in the number of targets detected. The results in scenario α and β are shown in Tables I and II.

In Scenario α, HMA-SAR markedly excelled in comparison to both the MARL and conventional methodologies. As delineated in Table I, given two agents operating within a constraint of 250 environmental steps, HMA-SAR achieved an SR of 97.61%, considerably exceeding those of HAPPO (82.07%), HAPPO-NC (80.48%), MASAC (76.49%), MAPPO (75.70%),

TABLE I
SCENARIO α: PERFORMANCE METRICS OF HMA-SAR AND BASELINES

Algorithm	SR (%)	ASC	VSC	AFT	VFT
HMA-SAR	97.61	125.91	2441.30	5.96	0.13
HAPPO	82.07	173.37	3610.95	5.62	0.86
HAPPO-NC	80.48	177.62	3990.81	5.65	0.75
MASAC	76.49	195.01	2772.69	5.57	0.85
MAPPO	75.70	188.49	3128.96	5.53	0.97
Frontier	23.90	213.26	5456.46	3.68	3.71

TABLE II
SCENARIO β: PERFORMANCE METRICS OF HMA-SAR AND BASELINES

Algorithm	SR (%)	ASC	VSC	AFT	VFT
HMA-SAR	80.88	185.04	2458.37	5.68	0.58
HAPPO	55.78	213.29	2300.06	4.71	2.14
HAPPO-NC	39.84	227.97	1502.94	5.08	1.51
MASAC	25.90	239.74	675.17	4.10	2.35
MAPPO	22.31	242.33	460.12	4.04	2.28
Frontier	1.20	249.57	240.52	0.96	2.38

and the frontier method at 23%. Importantly, while HMA-SAR has the lowest ASC and highest AFT, indicating HMA-SAR finds more targets using less time, the corresponding variance VSC and VFT in HMA-SAR are smaller than others, presenting that HMA-SAR is more robust facing various maps.

In Scenario β, as shown in Table II, the comparison results echo those found in Table I. Despite an overall decrease in SR, HMA-SAR stands out with an impressive SR of 80.88%, substantially higher than HAPPO (55.78%), HAPPO-NC (39.84%), MASAC (25.90%), and MAPPO (22.31%). Notably, the frontier-based exploration only achieved a 1.2% SR, underscoring its inefficacy. This emphasizes the challenges map coverage algorithms face, particularly with dynamic targets whose unpredictable movements allow them to repeatedly evade detection and return to previously explored areas. The ineffectiveness of map coverage methods in dynamic adaptation becomes evident as they require unnecessary revisiting of already explored areas to locate elusive targets. Similar to scenario α, HMA-SAR owns the lowest ASC and highest AFT, indicating HMA-SAR find more targets in less time, and the corresponding variance VSC and VFT in HMA-SAR is smaller than other MARL algorithms, presents that HMA-SAR is more robust facing various maps.

C. Scalability Study

In our study, we have specifically assessed the scalability of HMA-SAR by examining its SR and ASC metrics in response to escalating agent group sizes, ranging from 2 to 5. For each group size, we have kept the environment setting as Section VI-A and conducted these experiments under two distinct scenarios.

Fig. 8 delineates the trend in SR and ASC as agent numbers rise. Notably, the SR peaks at 100% from an initial 97.61% when the agent count reaches 4 and maintains this efficacy with 5 agents. Concurrently, ASC showcases a decline, moving from 125.91 to 81.65 as the agent numbers swell from 2 to 5. Fig. 9 showcases trends analogous to those previously observed in Fig. 8, with the SR curve ascending and the ASC curve descending. Specifically, SR reaches its zenith at 99.60% when there are 5 agents, while the ASC values achieve their nadir, registering at 110.42, also with 5 agents.

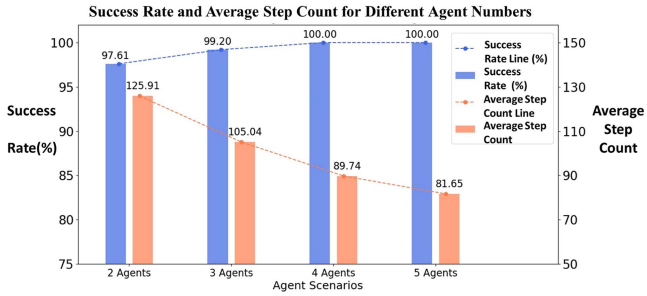

Fig. 8. **Scenario** α: Trend of SR and ASC as the agent number increases.

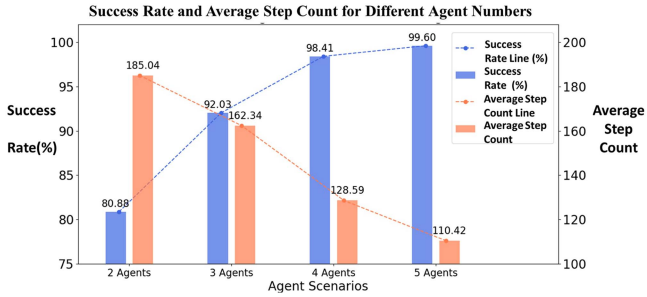

Fig. 9. **Scenario** β: Trend of SR and ASC as the agent number increases.

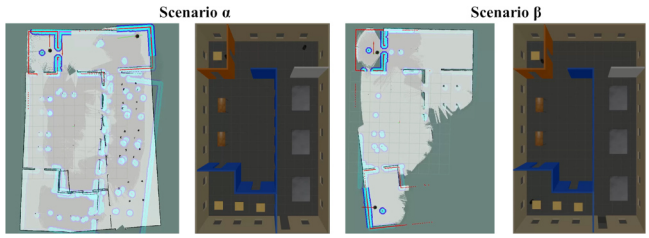

Fig. 10. Rviz and Gazebo results in scenarios α and β.

D. Experiments in Gazebo and Real-World Environment

In our experiments using Gazebo and real-world settings, we tested scenarios α and β. The Gazebo experiments included two TurtleBots and introduced three walking men as targets, where any human figure entering the detection range was immediately removed from the experiment's target list. The walking men were initiated from random positions following random trajectories. In scenario α, human figures were detectable right from the start, while in β, they remained undetected for the first 30 steps, to simulate evasion due to movement. Results depicted in Fig. 10 illustrate that in scenario α, TurtleBots achieved target detection by covering the entire map, whereas in β, they demonstrated a balance between exploration and search, without needing to cover the entire map.

Further insights were gained from Gridmap and Gazebo experiments. Agents trained under scenario α developed a strategy for comprehensive map coverage, while those in scenario β became adept at detecting overlooked targets. Subsequently, in the real-world experiments, we replicated these scenarios: In scenario α, to demonstrate the capability of complete map coverage, no human figures were included in the environment. In contrast, scenario β featured six human figures with colourful

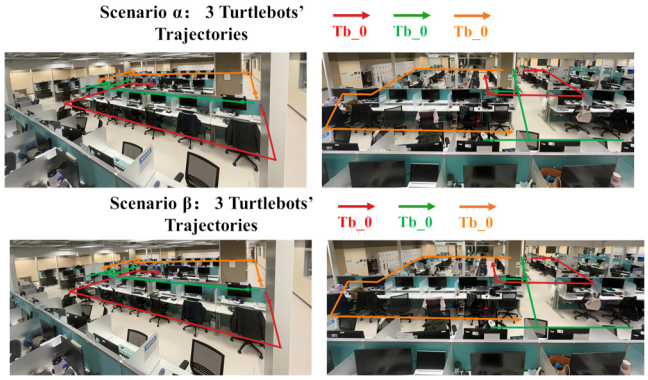

Fig. 11. Real-world experiments with three TurtleBots: Colored lines indicate their trajectories, with dual perspectives shown on the left and right of the same horizontal plane.

boards as dynamic targets. First, three walking men with colourful boards began their random movements from random starting points. Once all were detected, they reset and restarted their random movements from new positions within already explored areas, thereby stepping into roles as potentially missed targets. They would be detected by the TurtleBot's Yolo algorithm once within its detection range. The final trajectories are shown in Fig. 11. More details can be found in the video linked at https://youtu.be/0yFJ-mrWTvc.

VII. Conclusion

In this letter, we introduced a multi-agent search and rescue problem that has received limited attention in the literature. We presented a novel multi-agent SAR method, HMA-SAR, which incorporates unique state and reward designs, algorithmic constructs, and a hybrid decision mechanism. Comparative studies demonstrated that our HMA-SAR surpasses the performance of both frontier-based methods and other MARL algorithms. Looking forward, our future endeavours will focus on 1) enabling HMA-SAR to accommodate maps of arbitrary sizes, thereby transcending the constraints of maximum map size; and 2) training a universal model adaptable to a range of agent group sizes.

References

[1] Y. Liu and G. Nejat, "Multirobot cooperative learning for semiautonomous control in urban search and rescue applications," *J. Field Robot.*, vol. 33, no. 4, pp. 512–536, 2016.
[2] C. Luo, A. P. Espinosa, D. Pranantha, and A. De Gloria, "Multi-robot search and rescue team," in *Proc. IEEE Int. Symp. Saf. Secur. Rescue Robot.*, 2011, pp. 296–301.
[3] Z. Mou, Y. Zhang, F. Gao, H. Wang, T. Zhang, and Z. Han, "Deep reinforcement learning based three-dimensional area coverage with UAV swarm," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 10, pp. 3160–3176, Oct. 2021.
[4] F. Niroui, B. Sprenger, and G. Nejat, "Robot exploration in unknown cluttered environments when dealing with uncertainty," in *Proc. IEEE Int. Symp. Robot. Intell. Sensors*, 2017, pp. 224–229.
[5] R. Simmons et al., "Coordination for multi-robot exploration and mapping," in *Proc. 17th Nat. Conf. Artif. Intell., 12th Conf. Innov. Appl. Artif. Intell.*, 2000, pp. 852–858.
[6] B. A. Asfora, J. Banfi, and M. Campbell, "Mixed-integer linear programming models for multi-robot non-adversarial search," *IEEE Robot. Automat. Lett.*, vol. 5, no. 4, pp. 6805–6812, Oct. 2020.
[7] H. Lau, S. Huang, and G. Dissanayake, "Probabilistic search for a moving target in an indoor environment," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2006, pp. 3393–3398.
[8] G. Hollinger, S. Singh, J. Djugash, and A. Kehagias, "Efficient multi-robot search for a moving target," *Int. J. Robot. Res.*, vol. 28, no. 2, pp. 201–219, 2009.
[9] W. Luo and K. Sycara, "Adaptive sampling and online learning in multi-robot sensor coverage with mixture of Gaussian processes," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 6359–6364.
[10] C. Wakilpoor, P. J. Martin, C. Rebhuhn, and A. Vu, "Heterogeneous multi-agent reinforcement learning for unknown environment mapping," 2020, *arXiv:2010.02663*.
[11] 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. Automat. Lett.*, vol. 7, no. 2, pp. 3435–3442, Apr. 2022.
[12] A. H. Tan, F. P. Bejarano, Y. Zhu, R. Ren, and G. Nejat, "Deep reinforcement learning for decentralized multi-robot exploration with macro actions," *IEEE Robot. Automat. Lett.*, vol. 8, no. 1, pp. 272–279, Jan. 2023.
[13] W. Sheng, H. Guo, W.-Y. Yau, and Y. Zhou, "PD-FAC: Probability density factorized multi-agent distributional reinforcement learning for multi-robot reliable search," *IEEE Robot. Automat. Lett.*, vol. 7, no. 4, pp. 8869–8876, Oct. 2022.
[14] 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. Automat. Lett.*, vol. 4, no. 2, pp. 610–617, Apr. 2019.
[15] C. D. Koning and A. Jamshidnejad, "Hierarchical integration of model predictive and fuzzy logic control for combined coverage and target-oriented search-and-rescue via robots with imperfect sensors," *J. Intell. Robot. Syst.*, vol. 107, no. 3, 2023, Art. no. 40.
[16] M. Andries and F. Charpillet, "Multi-robot taboo-list exploration of unknown structured environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 5195–5201.
[17] Y. Gabriely and E. Rimon, "Spiral-STC: An on-line coverage algorithm of grid environments by a mobile robot," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2002, pp. 954–960.
[18] R. Arnold, J. Jablonski, B. Abruzzo, and E. Mezzacappa, "Heterogeneous UAV multi-role swarming behaviors for search and rescue," in *Proc. IEEE Conf. Cogn. Comput. Aspects Situation Manage.*, 2020, pp. 122–128.
[19] F. Chen, S. Bai, T. Shan, and B. Englot, "Self-learning exploration and mapping for mobile robots via deep reinforcement learning," in *Proc. AIAA Scitech Forum*, 2019, Art. no. 0396.
[20] Y. Yan et al., "Relative distributed formation and obstacle avoidance with multi-agent reinforcement learning," in *Proc. Int. Conf. Robot. Automat.*, 2022, pp. 1661–1667.
[21] Z. Li, J. Xin, and N. Li, "Autonomous exploration and mapping for mobile robots via cumulative curriculum reinforcement learning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2023, pp. 7495–7500.
[22] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, "Curiosity-driven exploration by self-supervised prediction," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 2778–2787.
[23] I. O. Tolstikhin et al., "MLP-mixer: An all-MLP architecture for vision," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 24261–24272.
[24] J. G. Kuba et al., "Trust region policy optimisation in multi-agent reinforcement learning," 2021, *arXiv:2109.11251*.
[25] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," 2015, *arXiv:1506.02438*.
[26] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6382–6393.
[27] C. Yu et al., "The surprising effectiveness of PPO in cooperative multi-agent games," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, pp. 24611–24624.
[28] B. Yamauchi, "Frontier-based exploration using multiple robots," in *Proc. 2nd Int. Conf. Auton. Agents*, 1998, pp. 47–53.