2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

MAexp: A Generic Platform for RL-based Multi-Agent Exploration

Shaohao Zhu, Jiacheng Zhou, Anjun Chen, Mingming Bai, Jiming Chen, and Jinming Xu

Abstract— **The sim-to-real gap poses a significant challenge in RL-based multi-agent exploration due to scene quantization and action discretization. Existing platforms suffer from the inefficiency in sampling and the lack of diversity in Multi-Agent Reinforcement Learning (MARL) algorithms across different scenarios, restraining their widespread applications. To fill these gaps, we propose MAexp, a generic platform for multi-agent exploration that integrates a broad range of state-of-the-art MARL algorithms and representative scenarios. Moreover, we employ point clouds to represent our exploration scenarios, leading to high-fidelity environment mapping and a sampling speed approximately 40 times faster than existing platforms. Furthermore, equipped with an attention-based Multi-Agent Target Generator and a Single-Agent Motion Planner, MAexp can work with arbitrary numbers of agents and accommodate various types of robots. Extensive experiments are conducted to establish the first benchmark featuring several high-performance MARL algorithms across typical scenarios for robots with continuous actions, which highlights the distinct strengths of each algorithm in different scenarios.**

I. Introduction

Multi-agent exploration is a rapidly growing field with various applications, such as search and rescue [1] and environmental surveillance [2]. Despite its promising perspective, developing coordination policies that perform efficiently across diverse scenarios remains a challenge. Traditional methods, such as Potential Field-Based [3] and Cost-Based Exploration [4], [5], commonly suffer from the limited representational capacity for coordination policies. Their efficiency can be sensitive to specific conditions [6] or require tedious parameter tuning [7] tailored for individual scenarios, highlighting the requirement for more robust approaches.

Recent advancements in Multi-Agent Reinforcement Learning [8], [9] have opened up avenues for improving the performance of multi-agent exploration. MARL algorithms have demonstrated their superior exploration efficiency over traditional methods in specific scenarios, such as indoor visual navigation [10], [7] and exploration in discrete grid environments [11], [6], [12]. However, these methods still exhibit a significant sim-to-real gap due to various factors, such as action discretization [10], scene quantization [11], [6], [12], and the neglect of physical collision constraints [7].

The simulation scenarios utilized for MARL training and execution play a key role in bridging the sim-to-real gap [13]. Different scenarios can provide diverse experiences, which is of great importance for robust strategies. On the other hand, the discrepancy between simulation scenarios and actual conditions directly influences the performance of derived strategies applied in the real world. However, most of the existing works rely heavily on non-standardized and randomly generated grid maps [14], [6], [11], which not only widens the sim-to-real gap but also limits the reproducibility and leads to unfair algorithmic comparisons. Other works utilizing open-source platforms with standardized maps for training and policy evaluation further encounter computational bottlenecks [13], [7], resulting in extensive training time, even with powerful hardware. Due to the tedious training process of MARL, the performance of various MARL algorithms in different exploration scenarios remains unexplored. This is an important issue to be solved to enable fair comparison among algorithms.

In this paper, we propose the first generic highly efficient Multi-Agent exploration platform (MAexp) that integrates a variety of MARL algorithms and scenarios. To narrow the sim-to-real gap, we leverage the benefits of point-cloud representations over traditional grid-based methods, dynamically adjusting point-cloud density to represent diverse exploration areas. This design allows for high-fidelity mapping in intricate regions while maintaining computational efficiency with less complex landscapes. To allow our agent framework to handle teams of arbitrary size, and to train exploration policies for any type of robot, we formulate our exploration problem into a two-step procedure: i) generating navigation goals for agents using a global attention-based target generator; ii) calculating a tailored navigation path to the goal via a local motion planner compatible with any navigation algorithms.

Our primary contributions are three-folds:

- We propose a generic high-efficiency platform for RL-based multi-agent exploration, accommodating various algorithms and scenarios, and achieving a sampling speed nearly 40 times faster than existing platforms.
- We employ an agent framework within the MAexp platform that can adapt to arbitrary team sizes and robot types during training.
- We establish a benchmark featuring six SOTA MARL algorithms and six typical scenarios, setting a foundational standard for rigorous evaluation and comparison of multi-agent exploration techniques.

II. Related Works

A. Reinforcement Learning in Multi-Agent Exploration

The existing frameworks for MARL in multi-agent exploration can be divided into two categories: Centralized

†The authors are with the College of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China. Correspondence to jimmyxu@zju.edu.cn (Jinming Xu). This work was supported in part by NSFC under Grants 62088101, 62373323 and in parts by the Key Laboratory of Collaborative Sensing and Autonomous Unmanned Systems (Key Lab of CS&AUS) of Zhejiang Province. Code and videos can be found at https://github.com/DuangZhu/MAexp.

Training with Centralized Execution (CTCE) [12], [15] and Centralized Training with Decentralized Execution (CTDE) [6], [7], [14], [16]. CTCE models, such as [15], employ attention mechanisms in MARL for grid-based exploration. [12] introduces CMAPPO, a CNN-augmented Proximal Policy Optimization variant, outperforming traditional frontier-based methods. However, CTCE is vulnerable to single-point failures and scalability issues because of its centralized architecture. CTDE methods are thus employed to address these limitations. For instance, [16] proposes a novel distributed exploration algorithm using Multi-Agent Deep Deterministic Policy Gradients (MADDPG) [17] for context-aware decision-making in structured settings. Subsequent studies, such as [7], enhance this by incorporating attention mechanisms for spatial and inter-agent information processing, while [6] extends CTDE frameworks to tackle decentralized exploration in complex terrains.

However, the current research usually narrows its focus to individual MARL algorithms and specific exploration scenarios, thereby limiting both cross-algorithmic and cross-scenario evaluation. Conversely, our study leverages six SOTA MARL algorithms across six different exploration scenarios to establish the first comprehensive benchmark, filling the notable gap in MARL exploration.

B. Existing Platforms for Multi-Agent Exploration

In the field of reinforcement learning based multi-agent exploration, platforms generally fall into two categories: vision-based exploration platforms using Habitat [18], as employed by MAANS [7], and grid-based platforms with discrete maps [6], [12], [13], [14]. While Habitat offers realistic simulations, it also involves additional modules, such as the SLAM module, which increases the sampling time during policy training. On the other hand, grid-based platforms suffer from a significant sim-to-real gap due to the substantial difference between the simulation scenarios and the real world. Moreover, most existing platforms support only a single MARL algorithm [7], [13], making cross-algorithmic comparisons impossible. In contrast, our platform provides a range of standard continuous exploration scenarios while offering multiple MARL algorithms for proper comparisons. It utilizes high-fidelity point-cloud-based scenarios and continuous actions to narrow the sim-to-real gap, and enhances computational efficiency for faster sampling. Note that the agent framework in our platform, built on the foundation of MARLLib [19], is integrated with a local planner to account for various types of robots.

III. Problem Formulation

A. Task Setup

We investigate a multi-agent coordination exploration problem, focusing on a team of robots tasked with cooperatively exploring an unknown continuous scenario as swiftly as possible, as illustrated in Fig. 1. The explored areas are highlighted in green, with unexplored areas in yellow. Utilizing the Ackermann Car Model as a representative example, each agent performs two continuous actions: Acceleration and Steering Angle. At each time step, the agent acquires point-cloud data (shown as green points within radar range) from sensors like radars or cameras. In our framework, we assume perfect communication between agents, allowing for the exchange of observations, states, and goals to generate actions during both the training and execution phases. The primary objective of this task is to maximize the accumulated explored area (entire green region) while minimizing the overlap (indicated by the blue area) between agents within a constrained time horizon.

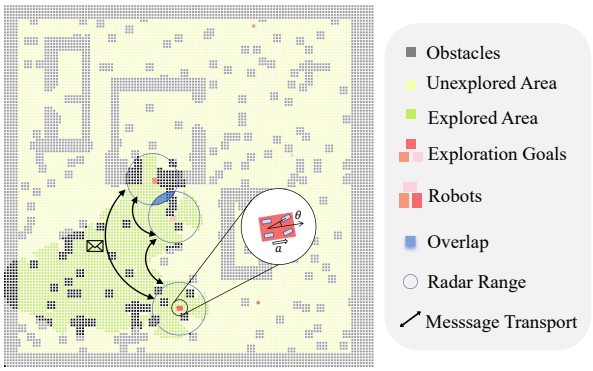

Fig. 1. Multi-agent coordination exploration in a random obstacle scenario.

B. Mathematical Formulation

While implementing MARL algorithms, it is essential to model the problem as a decentralized partially observable Markov decision process (Dec-POMDP). A Dec-POMDP is defined by the tuple $\langle n, S, A, O, R, P, \gamma, h\rangle$. Here, n denotes the number of agents involved in the task, and S signifies the state space, with s_i representing the state of agent i and s representing the joint state of all agents. The joint action space is symbolized by A, and the individual actions of agent i and joint action are represented by a_i and a. An observation for agent i, denoted by o_i, is generated by the observation function $O(s, a_i)$, dependent on both the states of all agents and their corresponding actions. The reward for agent i and the whole team, symbolized by r_i and r, can be derived from the reward function $R(s, a)$. The transition probability from state s to state s' through action a is defined by $P(s'|s, a)$, dependent on the environment's properties. The discount factor for future rewards is represented by γ, and h indicates the horizon of an episode. The goal of this problem is for each agent to find the most effective strategy, denoted as π_i, and collaboratively work towards maximizing the collective reward for the team as follows:

$$J(\theta) = \mathbb{E}\left[\sum_{t=0}^{h-1} \gamma^t R(s(t), a(t)) \middle| s(0), \pi\right].$$

The ideal coordination strategy is formulated using gradient descent to optimize the objective function $J(\theta)$ relative to π.

C. Metrics

To evaluate exploration policies, we divide our metrics into two dimensions: Team-Based and Agent-Specific metrics.

These metrics evaluate both the efficiency and cooperation abilities of the team within the exploration process.

1) Team-Based Metrics:

- **Exploration Ratio (ER):** Quantifies the proportion of the explored area to the total explorable space at the end of an episode.
- **Coverage Step (CS):** Denotes the number of steps required for the team to reach the predefined exploration thresholds (85% and 95%). An 85% ratio implies significant topological coverage, while 95% marks the episode's successful completion.

2) Agent-Specific Metrics:

- **Mutual Overlap (MO):** Measures the ratio of the area only explored by one agent to the total explored area. Less overlap suggests better task allocation and minimized redundancy. This metric is recorded when the exploration ratio reaches 85% and 95%.
- **Reward standard Variance (RV):** Captures the variability in rewards across agents at the end of an episode. A lower variance indicates a more balanced contribution from each agent, avoiding situations where certain agents underperform but are overshadowed by the team's collective results.

D. Reward Function

Our reward function uniquely fuses team performance with agent contributions, comprising five elements: Success Reward, Exploration Reward, Overlap Penalty, Collision Penalty, and Time Penalty. Let Cov_t represent the total coverage ratio at time t, map_t^{tot} the merge map, and map_t^i the area explored by agent i at the same time step. In all map variables, '1' denotes explored space, while '0' indicates unexplored territory.

The agent-specific total reward R^i_{total} is formalized as:

$$R^i_{\text{total}} = \begin{cases} R_s, & \text{(Success Reward)} \\ \delta map_t^i, & \text{(Exploration Reward)} \\ -\sum(P_t^{-i} \cap P_t^i), & \text{(Overlap Penalty)} \\ -R_c, & \text{(Collision Penalty)} \\ -Cov_t, & \text{(Time Penalty)} \end{cases}$$

Here, δmap_t^i is proportional to $\sum(map_t^i - map_{t-1}^{tot})$, where the summation is over all locations where the difference equals 1. P_t^i and P_t^{-i} denote the point clouds gathered by agent i and the rest of the team at time t, respectively. Note that the final agent-specific reward is a normalized linear combination of these five components.

IV. The Proposed Platform

A. The proposed MAexp Platform

We introduce MAexp, a generic high-efficiency platform designed for multi-agent exploration, encompassing a diverse range of scenarios and MARL algorithms. The platform is developed in Python to smoothly integrate with existing reinforcement learning algorithms, and it is equally applicable to traditional exploration methods. In an effort to bridge the sim-to-real gap, all maps and agent properties within MAexp are modelled continuously, incorporating realistic physics to closely mirror real-world exploration.

The autonomous exploration simulation employs an integrated data flow, as shown in Fig. 2 (a-b). Part (b) presents the platform's sampling mechanism. In this phase, MAexp initiates multiple environments to collect vast experience. Agents, at every time step, acquire point-cloud data and their respective states from the environment. Moreover, they can adjust the radar resolution and detection range based on specific situations. The point-cloud data generation during simulation is depicted in Fig. 2 (c). This procedure simulates radar perception but operates at a faster speed. Once an agent completes an action, the environment identifies all free space point clouds within the detection range of the agent. Subsequent filtering eliminates points masked by obstacles using the following criteria:

$$\begin{cases} \frac{P_f \cdot P_o}{\|P_f\|\|P_o\|} > 1 - \alpha_1; \\ \|P_f\| - \|P_o\| < \alpha_2, \end{cases}$$

where P_f denotes the vector from the robot to a free space point, P_o represents the vector to an obstacle, and α_1, α_2 are certain parameters to be determined. This process filters out obstructed points (c.f., blue points in Fig. 2 (c)) and transmits the remaining ones to the agent. Utilizing past feature maps and environmental data, the current feature map is generated and distributed to all agents within the environment. The agent frameworks consequently produce the actions for agents. Implementing these actions provides individual rewards for every agent, mirroring the action value. Thereafter, the environment transitions the collective agents to the succeeding states. MAexp, through consistent sampling, MAexp accumulates a substantial set of quadruples $\langle s, a, r, s' \rangle$ for training policies. Fig. 2 (a) illustrates the training process and provides an in-depth examination of the Multi-Agent Target Generator in the agent framework. During training, batches of quadruples are transported to both the actor and the critic to facilitate parameter updating. The critic provides the actor with the action values, which are essential for the gradient descent step.

It's worth noting that the proposed agent framework is well-adapted to a range of robots and group sizes. In particular, we address the multi-agent exploration challenge leveraging the new techniques developed for robotic navigation [20], [21], [22], [23]. Therefore, we divide our agent framework into two modules: *Multi-Agent Target Generator (where-to-go)* and *Single-Agent Motion Planner (how-to-go)*. In the first module, agents use MARL algorithms to generate their own navigation points. In the second module which is independent of MARL, agents decide how to arrive these target locations by determining their acceleration and steering angle. Notably, the how-to-go module can incorporate any robot navigation algorithm for motion planning.

B. Scenarios in MAexp

Different from previous studies that rely on grid-based scenarios, we utilize point clouds for all map formulations.

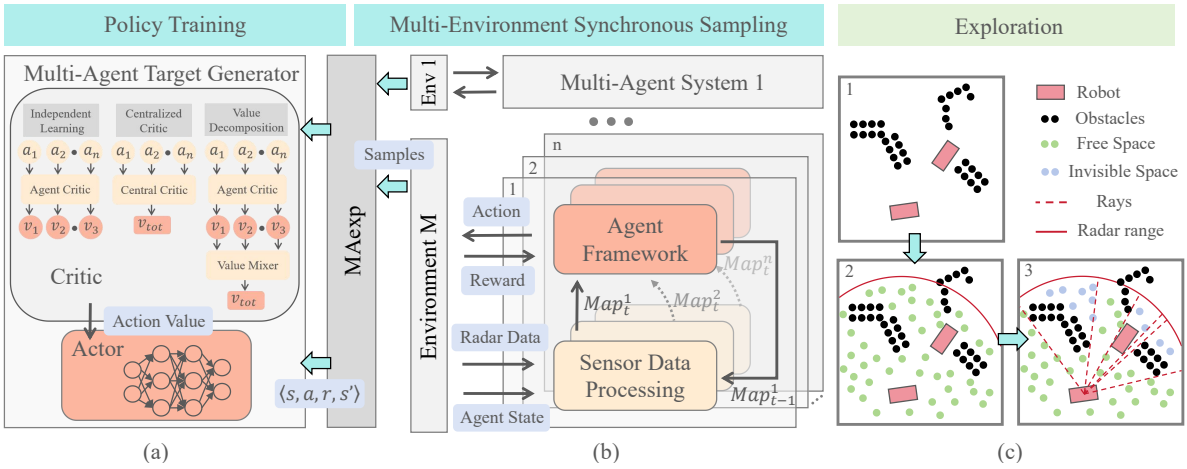

Fig. 2. The proposed MAexp platform for multi-agent exploration.

This method ensures a seamless depiction of exploration areas, delivering both detailed and authentic representations. The processing of point clouds is easily aligned with GPU parallel processing, and the map's sparsity does not require global consistency. In obstacle-dense regions, for instance, there's no need to detail every obstacle—defining boundaries suffices. In pivotal regions, increasing the density of the point cloud can bolster map authenticity. Furthermore, point-cloud maps can be effortlessly extended into three-dimensional spaces. The detailed characteristics of the four unique exploration scenarios in MAexp, along with their 3/2D maps, can be found in TABLE I and Fig. 3.

TABLE I

SUMMARY OF VARIOUS MULTI-AGENT EXPLORATION SCENARIOS.

Properties		Type	Resolution	Size(m)	Quantity
Random Obstacle		SPC	1.0	125	32
Maze		SPC	1.0	125	80
Indoor	Small	RPC	0.1	<10	17
	Medium	RPC	0.1	10-14	32
	Large	RPC	0.1	>14	22
Outdoor		SPC & RPC	1.5-2	288	32

*RPC: Real-world Point Clouds; SPC: Synthetic Point Clouds.

Random Obstacle scenarios combine basic elements like narrow corridors, corner loops, and multiple rooms, as detailed in [13]. Then we intersperse isolated obstacles until the map achieves the desired obstacle density. This scenario is specifically designed to encapsulate a diverse range of exploration difficulties commonly faced by agents.

Maze scenarios are generated through a refined Kruskal's algorithm. With numerous branching paths and the absence of closed loops, the Maze necessitate agents to strategically allocate different agents to distinct paths to improve exploration efficiency.

Indoor scenarios are derived from Habitat [18], a high-fidelity dataset of 3D indoor scans. By closely mirroring actual indoor settings, these scenarios challenge the agent to conduct thorough exploration in constrained areas. Based on the map size, we categorize these scenarios into three types: large, medium, and small.

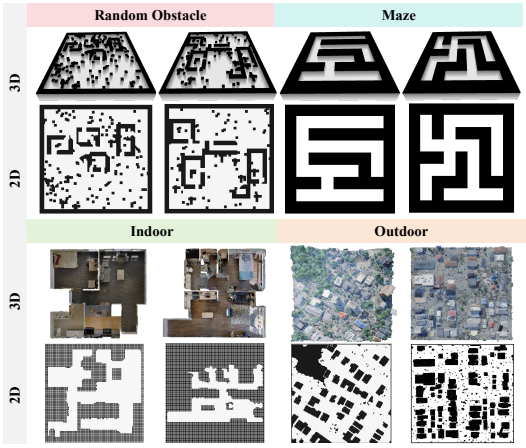

Fig. 3. Illustration of exploration scenarios in MAexp with 3D visualizations and 2D top-down projection maps.

Outdoor scenarios are derived from STPLS3D [24], which comprises a comprehensive collection of synthetic and actual aerial photogrammetry 3D point clouds. These maps authentically emulate extensive outdoor exploration settings, thereby challenging the system's adeptness at swift, large-scale outdoor exploration.

It should be noted that during the process of map creation, we conducted careful checks and repairs to ensure the maps were suitable for exploration tasks, which is time-consuming but essential in achieving accurate results.

C. MARL Algorithms in MAexp

MARL algorithms can be broadly classified into three categories based on their critic architectures: Independent Learning (IL), Centralized Critic (CC), and Value Decomposition (VD). These architectures are illustrated in Fig. 2(a).

Independent Learning such as IPPO [25], IA2C [26], and ITRPO [27], operate with agent-level critic, focusing on singular observations. IPPO, for instance, exhibits robustness to certain environmental variability [28], while ITRPO dom-

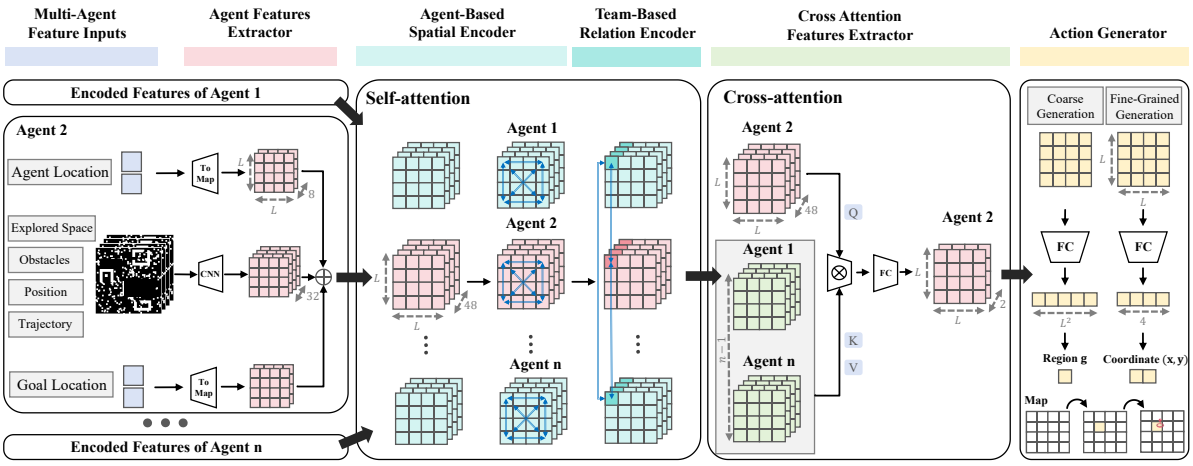

Fig. 4. The overall structure of Multi-agent Target Generator.

inates in the MAMuJoCo [29] benchmark for multi-agent robotic control [19].

Centralized Critic, including MAPPO [30] and MATRPO [31], employ a unified critic that fuses information from the entire team to judge the joint action value. Generally, CC algorithms outperform IL in coordinated tasks [19], [30].

Value Decomposition, including VDPPO [32] and VDA2C [33], combine outputs from agent-level critics to estimate a team action value, thereby balancing both individual and group objectives. Despite their best performance in numerous benchmarks [8], [17], [19], they struggle with continuous control and long-term planning, typically seen in multi-agent exploration [19].

Recognizing the unique strengths of individual algorithms, we have incorporated into the proposed MAexp platform six leading MARL algorithms, such as IPPO, ITRPO, MAPPO, MATRPO, VDPPO, and VDA2C.

D. Multi-Agent Target Generator

The *Multi-agent Target Generator*, whose entire structure is illustrated in Fig. 4, employs a CNN model to extract 125×125 grid-based spatial features from each agent, with inputs capturing explored space, obstacles, agent position, and trajectory. These feature maps, when combined with embedded agent and previous goal location, yield an $L \times L \times D$ feature map where $D = 48$. Subsequently, team-wise data is fused via a hierarchical transformer-based structure, similar to *Spatial-TeamFormer* in MAANS [7], known for its superior performance in visual exploration. The *Spatial-Teamformer block* integrates two layers: the *Individual Spatial Encoder*, which applies spatial self-attention to each agent's $L \times L \times D$ map without inter-agent computations, and the *Team Relation Encoder*, targeting team interactions without spatial considerations. These team-oriented data are then merged for each agent through cross-attention. In our approach, the transformer depth is fixed at 2, yielding a $2 \times L \times L$ grid feature as the output.

The $2 \times L \times L$ grid feature feeds a dual-action generator to produce an exploration target. Using its first channel, the agent determines a discrete region for coarse generation. This feature translates to an L^2 vector, denoting the probability of each region's selection, from which a region is sampled. Subsequently, for fine-grained generation, a coordinate (x, y) denotes the global goal's relative position within the chosen region. This coordinate is derived from a 4-vector, where the initial pair indicates the mean and variance for coordinate x, and the latter pair for y. Note that the above two generators operate in parallel and thus it is not necessary to feed the coarse action into the fine-grained generator.

V. Experiments

A. Environmental Setup

In our experimental studies, we employ a swarm of Ackermann cars equipped with adjustable radars, tailoring the resolution and detection range to various exploration scenarios. The *Multi-Agent Target Generator* operates as described earlier, while the *Single-Agent Motion Planner* employs the Dynamic Window Approach (DWA), which provides precise navigation and obstacle avoidance.

Our experiments involve $N = 3$ agents, with parameters set at $L = 8$, $\alpha_1 = 10^{-3}$, $\alpha_2 = 10^{-6}$, $R_s = 100$, and $R_c = 200$. Each MARL training spans 10^4 iterations across three random seeds. Results are presented as "mean (standard deviation)," averaged from 300 tests—100 per seed. We apply all six MARL algorithms across 16 settings: 4 Random Obstacle, 4 Maze, 3 Outdoor, and 5 Indoor. We further classify the "Indoor" into 2 small, 2 medium, and 1 large. Training is carried out on an Ubuntu 18.04 server with two NVIDIA GeForce RTX 3090 GPUs: one for multi-environment sampling and another for online training. Each exploration map receives specialized parameter training to optimize algorithm performance. A 10^4 iteration training requires about 60 hours, culminating in a total of roughly 750 days of continuous GPU usage across over 300 runs.

B. Comparison of Simulation Speed

We first compare the sampling speeds of several open-source SOTA platforms for MARL exploration, i.e., MAexp, MAANS [7], and Explore-Bench [13].

TABLE II

PERFORMANCE RESULTS OF SIX MARL ALGORITHMS ACROSS SIX SCENARIOS. WITHIN EACH SCENARIO, THE UPPER THREE COLUMNS REPRESENT 'ER' (↑), '85% CS' (↓), AND '95% CS' (↓), WHILE THE LOWER COLUMNS CORRESPOND TO '85% MO' (↓), '95% MO' (↓), AND 'RV' (↓).

Scenarios		Generated Scenes						Real Scenes											
		Random Obstacles			Maze			Indoor-Small			Indoor-Medium			Indoor-Large			Outdoor		
		ER (%)	85%CS (step)	95%CS (step)	ER (%)	85%CS (step)	95%CS (step)	ER (%)	85%CS (step)	95%CS (step)	ER (%)	85%CS (step)	95%CS (step)	ER (%)	85%CS (step)	95%CS (step)	ER (%)	85%CS (step)	95%CS (step)
Team-Based	ITRPO	**68.81** ± 5.71	-	-	79.71 ± 7.32	449 ± 63	501 ± 77	90.46 ± 5.12	268 ± 45	437 ± 65	79.38 ± 5.00	400 ± 84	557 ± 47	59.40 ± 3.95	-	-	26.91 ± 4.12	-	-
	IPPO	63.07 ± 4.69	-	-	**93.17** ± 4.27	377 ± 51	441 ± 57	**94.18** ± 0.76	**157** ± 5	**205** ± 41	**91.48** ± 4.36	**287** ± 54	470 ± 61	71.32 ± 4.79	-	-	33.06 ± 6.90	-	-
	MATRPO	61.12 ± 3.76	-	-	86.29 ± 7.63	376 ± 72	472 ± 65	89.66 ± 0.79	237 ± 13	356 ± 92	75.73 ± 8.91	472 ± 44	536 ± 67	**77.31** ± 4.39	-	-	32.72 ± 4.31	-	-
	MAPPO	62.53 ± 3.38	-	-	89.15 ± 5.07	370 ± 67	442 ± 44	93.23 ± 3.56	206 ± 88	284 ± 107	87.01 ± 5.71	356 ± 40	-	70.19 ± 4.94	-	-	**38.15** ± 7.24	-	-
	VDPPO	60.04 ± 4.54	-	-	83.12 ± 5.08	468 ± 45	506 ± 35	91.85 ± 3.01	236 ± 92	358 ± 111	89.97 ± 3.11	308 ± 41	**447** ± 24	67.72 ± 4.96	-	-	30.31 ± 5.48	-	-
	VDA2C	64.23 ± 3.43	-	-	**93.23** ± 3.54	**322** ± 59	**416** ± 54	**93.59** ± 0.33	170 ± 12	235 ± 52	76.93 ± 3.95	320 ± 30	**449** ± 26	73.97 ± 2.86	-	-	26.76 ± 4.38	-	-
		85%MO (%)	95%MO (%)	RV	85%MO (%)	95%MO (%)	RV	85%MO (%)	95%MO (%)	RV	85%MO (%)	95%MO (%)	RV	85%MO (%)	95%MO (%)	RV	85%MO (%)	95%MO (%)	RV
Agent-Specific	ITRPO	-	-	**610.73** ± 79.25	51.56 ± 6.49	43.89 ± 4.76	931.00 ± 104.18	**51.37** ± 4.84	**48.90** ± 2.54	871.78 ± 38.41	49.60 ± 7.49	44.77 ± 6.84	347.76 ± 42.10	-	-	252.95 ± 104.43	-	-	457.93 ± 84.23
	IPPO	-	-	711.56 ± 109.97	**45.52** ± 7.87	**42.38** ± 7.12	970.43 ± 76.31	54.06 ± 3.55	50.89 ± 4.59	576.31 ± 8.84	46.38 ± 3.35	46.39 ± 4.36	318.29 ± 50.87	-	-	257.30 ± 59.91	-	-	534.92 ± 94.25
	MATRPO	-	-	825.65 ± 63.06	48.21 ± 9.19	43.91 ± 8.12	910.37 ± 82.33	57.49 ± 4.59	59.37 ± 4.29	547.39 ± 45.81	49.92 ± 8.45	57.92 ± 8.39	389.70 ± 89.74	-	-	280.52 ± 74.28	-	-	433.80 ± 115.81
	MAPPO	-	-	831.93 ± 94.97	46.22 ± 9.46	38.53 ± 2.95	929.25 ± 89.14	60.78 ± 9.24	59.14 ± 8.47	602.14 ± 78.95	38.22 ± 2.81	-	385.22 ± 49.97	-	-	248.80 ± 87.86	-	-	445.98 ± 118.32
	VDPPO	-	-	**632.18** ± 83.41	58.96 ± 5.03	58.66 ± 4.02	951.11 ± 82.10	64.03 ± 15.29	66.83 ± 17.67	541.55 ± 99.39	**36.58** ± 1.63	42.38 ± 0.74	**309.89** ± 31.52	-	-	**198.38** ± 78.41	-	-	**431.67** ± 103.54
	VDA2C	-	-	810.62 ± 76.26	49.60 ± 5.65	47.62 ± 5.91	**907.74** ± 60.99	77.14 ± 3.89	77.32 ± 8.40	**496.54** ± 33.77	45.20 ± 3.32	**40.45** ± 1.04	445.24 ± 39.86	-	-	275.55 ± 37.27	-	-	561.50 ± 114.43

TABLE. III illustrates the sampling times per step for team sizes $N = 2, 4, 6, 8$. The proposed MAexp platform, optimized for MARL exploration, achieves a simulation speed nearly 40 times faster than MAANS. While both utilize the same exploration principles, MAANS, designed for vision tasks, incur additional computational costs. In contrast, Explore-Bench's level-0 component, similar to ours, is tailored for MARL sampling. Unlike Explore-Bench's CPU-centric grid simulations, MAexp leverages point-cloud modelling and GPU parallelization, substantially accelerating simulations. Hence, MAexp emerges as the most efficient platform for MARL exploration, facilitating the evaluation and development of new MARL algorithms. Note that our platform can also accommodate a large number of robots as long as communication and action generation strategies are properly adjusted for enhanced efficiency.

TABLE III

COMPARISON OF SAMPLING TIMES(S) FOR DIFFERENT PLATFORMS

	$N = 2$	$N = 4$	$N = 6$	$N = 8$
MAANS [7]	0.4497	0.9319	1.3675	1.9354
Explore-Bench [13]	3.8586	6.6381	8.6575	10.9880
MAexp (ours)	**0.0144**	**0.0254**	**0.0373**	**0.0477**

C. Performance of MARL Algorithms in Various Scenarios

In Table II, we observe that different algorithms exhibit different characteristics across our proposed scenarios. In particular, ITRPO performs impressively well in the "Random Obstacles" scenario, while IPPO consistently achieves the best exploration ratios in "Maze", "Indoor-Small," and "Indoor-Medium ". These IL-based approaches excel in small-scale exploration scenarios characterized by dense challenges such as corner loops and multiple rooms, as their critics primarily focus on the agent's immediate environment to generate suitable strategies.

VD-based algorithms, such as VDPPO and VDA2C, tend to obtain efficient exploration policies by uniform task distribution, especially when agents maintain consistent performance. This is evident in the consistently low "Reward Variance" observed in VDPPO and VDA2C across all scenarios, even though they exhibit lower exploration ratios. Further, the superior performance of VDA2C in "Maze" reinforces the perspective since the efficiency of exploration among agents becomes uniformly distributed in this structured environment with a constant width of free space.

In contrast, CC-based approaches, such as MATRPO and MAPPO, demonstrate superior performance in large-scale scenarios characterized by sparsely distributed obstacles as they tend to disperse agents to different regions of the map for parallel exploration. As evidenced in the Table, MATRPO achieves a 77.31% exploration ratio in "Indoor-Large" while MAPPO excels in "Outdoor". These algorithms allow for an overall understanding of the situation of the entire team by fusing the observations and states among agents, facilitating efficient spatial allocation. However, the fusion also increases the complexities of policy training, making it challenging to adopt strategies tailored to immediate surroundings, resulting in lower performance in small-scale scenarios.

To sum up, understanding the characteristics of the exploration scenario is crucial for choosing a proper MARL algorithm to generate robust and efficient coordination policies. Once the scenario is identified, the proposed MAexp platform offers a valuable tool for evaluating candidate MARL algorithms comprehensively, facilitating the selection of the most appropriate one. Moreover, for those involved in designing new algorithms, MAexp also serves as a generic, high-efficiency platform for both training and simulation, as well as a benchmark for performance comparison.

VI. CONCLUSIONS

We introduced MAexp, a generic high-efficiency platform for multi-agent exploration. MAexp incorporates several state-of-the-art MARL algorithms and various representative exploration scenarios, and it employs point-cloud representation for maps which enhances the effectiveness of MARL algorithms with rapid sampling and realistic simulation environments. Moreover, with its well-designed agent framework, MAexp can accommodate a variety of robots and group sizes. Furthermore, we establish the first comprehensive benchmark featuring several high-performance MARL algorithms across various typical scenarios. Our results highlight the unique strengths of each algorithm in different scenarios. In our future work, we aim to enhance MAexp to account for general communication topology and incorporate more advanced MARL algorithms and practical scenarios so as to provide a versatile platform for multi-agent exploration. We believe that our platform can advance the field of RL-based multi-agent exploration.

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] A. Fascista, "Toward integrated large-scale environmental monitoring using wsn/uav/crowdsensing: A review of applications, signal processing, and future perspectives," *Sensors*, vol. 22, no. 5, p. 1824, 2022.

[3] J. Yu, J. Tong, Y. Xu, Z. Xu, H. Dong, T. Yang, and Y. Wang, "Smmr-explore: Submap-based multi-robot exploration system with multi-robot multi-target potential field exploration method," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 8779–8785.

[4] Y. Mei, Y.-H. Lu, C. G. Lee, and Y. C. Hu, "Energy-efficient mobile robot exploration," in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.* IEEE, 2006, pp. 505–511.

[5] S. Oßwald, M. Bennewitz, W. Burgard, and C. Stachniss, "Speeding-up robot exploration by exploiting background information," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 716–723, 2016.

[6] 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 Robotics and Automation Letters*, vol. 8, no. 1, pp. 272–279, 2022.

[7] C. Yu, X. Yang, J. Gao, H. Yang, Y. Wang, and Y. Wu, "Learning efficient multi-agent cooperative visual exploration," in *European Conference on Computer Vision*. Springer, 2022, pp. 497–515.

[8] M. Samvelyan, T. Rashid, C. S. De Witt, G. Farquhar, N. Nardelli, T. G. Rudner, C.-M. Hung, P. H. Torr, J. Foerster, and S. Whiteson, "The starcraft multi-agent challenge," *arXiv preprint arXiv:1902.04043*, 2019.

[9] B. Baker, I. Kanitscheider, T. Markov, Y. Wu, G. Powell, B. McGrew, and I. Mordatch, "Emergent tool use from multi-agent autocurricula," *arXiv preprint arXiv:1909.07528*, 2019.

[10] H. Wang, W. Wang, X. Zhu, J. Dai, and L. Wang, "Collaborative visual navigation," *arXiv preprint arXiv:2107.01151*, 2021.

[11] A. Mete, M. Mouhoub, and A. M. Farid, "Coordinated multi-robot exploration using reinforcement learning," in *2023 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2023, pp. 265–272.

[12] Z. Chen, B. Subagdja, and A.-H. Tan, "End-to-end deep reinforcement learning for multi-agent collaborative exploration," in *2019 IEEE International Conference on Agents (ICA)*. IEEE, 2019, pp. 99–102.

[13] Y. Xu, J. Yu, J. Tang, J. Qiu, J. Wang, Y. Shen, Y. Wang, and H. Yang, "Explore-bench: Data sets, metrics and evaluations for frontier-based and deep-reinforcement-learning-based autonomous exploration," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 6225–6231.

[14] 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 Robotics and Automation Letters*, vol. 7, no. 2, pp. 3435–3442, 2022.

[15] 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, 2019.

[16] D. He, D. Feng, H. Jia, and H. Liu, "Decentralized exploration of a structured environment based on multi-agent deep reinforcement learning," in *2020 IEEE 26th International Conference on Parallel and Distributed Systems (ICPADS)*. IEEE, 2020, pp. 172–179.

[17] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *Advances in neural information processing systems*, vol. 30, 2017.

[18] A. Szot, A. Clegg, E. Undersander, E. Wijmans, Y. Zhao, J. Turner, N. Maestre, M. Mukadam, D. Chaplot, O. Maksymets, A. Gokaslan, V. Vondrus, S. Dharur, F. Meier, W. Galuba, A. Chang, Z. Kira, V. Koltun, J. Malik, M. Savva, and D. Batra, "Habitat 2.0: Training home assistants to rearrange their habitat," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[19] S. Hu, Y. Zhong, M. Gao, W. Wang, H. Dong, Z. Li, X. Liang, X. Chang, and Y. Yang, "Marllib: Extending rllib for multi-agent reinforcement learning," *arXiv preprint arXiv:2210.13708*, 2022.

[20] A. Sadek, G. Bono, B. Chidlovskii, A. Baskurt, and C. Wolf, "Multi-object navigation in real environments using hybrid policies," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 4085–4091.

[21] K. Nakhleh, M. Raza, M. Tang, M. Andrews, R. Boney, I. Hadžić, J. Lee, A. Mohajeri, and K. Palyutina, "Sacplanner: Real-world collision avoidance with a soft actor critic local planner and polar state representations," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 9464–9470.

[22] Y. Shu, Z. Li, B. Karlsson, Y. Lin, T. Moscibroda, and K. Shin, "Incrementally-deployable indoor navigation with automatic trace generation," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2019, pp. 2395–2403.

[23] Z. Zhang, S. He, Y. Shu, and Z. Shi, "A self-evolving wifi-based indoor navigation system using smartphones," *IEEE Transactions on Mobile Computing*, vol. 19, no. 8, pp. 1760–1774, 2019.

[24] M. Chen, Q. Hu, Z. Yu, H. THOMAS, A. Feng, Y. Hou, K. McCullough, F. Ren, and L. Soibelman, "Stpls3d: A large-scale synthetic and real aerial photogrammetry 3d point cloud dataset," in *33rd British Machine Vision Conference 2022, BMVC 2022, London, UK, November 21-24, 2022*. BMVA Press, 2022. [Online]. Available: https://bmvc2022.mpi-inf.mpg.de/0429.pdf

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

[26] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*. PMLR, 2016, pp. 1928–1937.

[27] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International conference on machine learning*. PMLR, 2015, pp. 1889–1897.

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

[29] B. Peng, T. Rashid, C. Schroeder de Witt, P.-A. Kamienny, P. Torr, W. Böhmer, and S. Whiteson, "Facmac: Factored multi-agent centralised policy gradients," *Advances in Neural Information Processing Systems*, vol. 34, pp. 12 208–12 221, 2021.

[30] C. Yu, A. Velu, E. Vinitsky, J. Gao, Y. Wang, A. Bayen, and Y. Wu, "The surprising effectiveness of ppo in cooperative multi-agent games," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24 611–24 624, 2022.

[31] J. G. Kuba, R. Chen, M. Wen, Y. Wen, F. Sun, J. Wang, and Y. Yang, "Trust region policy optimisation in multi-agent reinforcement learning," *arXiv preprint arXiv:2109.11251*, 2021.

[32] Y. Ma and J. Luo, "Value-decomposition multi-agent proximal policy optimization," in *2022 China Automation Congress (CAC)*. IEEE, 2022, pp. 3460–3464.

[33] J. Su, S. Adams, and P. Beling, "Value-decomposition multi-agent actor-critics," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 13, 2021, pp. 11 352–11 360.