Multi-agent Exploration with Similarity Score Map and Topological Memory

Eun Sun Lee[1] and Young Min Kim[1*]

***Abstract*—Multi-robot exploration can be a collaborative solution for navigating a large-scale area. However, it is not trivial to optimally assign tasks among agents because the state dynamically changes while the local observations of multiple agents concurrently update the global map. Furthermore, the individual robots may not have access to accurate relative poses of others or global layouts. We propose an efficient spatial abstraction for multi-agent exploration based on topological graph memories. Each agent creates a topological graph, a lightweight spatial representation whose nodes contain minimal image features. The information in graphs is aggregated to compare individual nodes and is used to update the similarity scores in real-time. Then, the agents effectively fulfill distributed task goals by examining the dynamic similarity scores of frontier nodes. We further exploit extracted visual features to refine the relative poses among topological graphs. Our proposed pipeline can efficiently explore large-scale areas among various scene and robot configurations without sharing precise geometric information. Code is available here.**

***Index Terms*—Path Planning for Multiple Mobile Robots or Agents, Vision-Based Navigation, Multi-Robot Systems, Topological Map.**

I. Introduction

AUTONOMOUS navigation is a fundamental process for fulfilling tasks in the intelligence of mobile robots. Specifically, we focus on an exploration task for intelligent control of mobile robots, which critically enhances the likelihood of target detection or task accomplishment. We further investigate multi-agent exploration, in which multiple robots, rather than a single isolated robot, collaborate to provide more comprehensive coverage of the environment without ground-truth poses. While single-agent exploration can be overwhelming in large-scale environments, multiple robots, if properly designed, can fulfill demand, improving productivity and resource utilization.

However, existing studies [1]–[4] on exploration approaches perform the task with a single agent and still are confined under trade-off for different algorithmic choices. The efficacy of exploration mainly depends on the global planner, which determines long-term goal positions for path planning. The prevailing literature [5]–[7] has employed the frontier-based method [8] as a global planner. An agent identifies frontiers, the regions marking the boundary between explored and unexplored areas, and explores them to strategically complete scene maps. Frontier selection relies on manually designed rules such as prioritizing the closest frontier over primary exploration goals. Recently, learning-based methods have emerged, training policies to select goals based on task reward function [1], [9], [10]. However, these policies, unlike frontier-based methods, lack explicit geometric reasoning and may not ensure a complete scene exploration. Furthermore, most learning-based methods are effective only in their training configurations, including actuators, sensors, and computational resources, limiting the deployment conditions. Map-based learning methods [11], [12], for instance, utilize a predefined size of a global map and cannot scale to large-scale environments without separate training with additional datasets and memory.

As we extend into a multi-robot scenario, two primary objectives emerge as additional critical challenges. First, the optimal reward needs to coordinate the individual goals for all agents. The overall exploration reward hinges on the collective actions or observations of all agents. Real-time collaboration between multiple agents demands dynamic role reconfiguration, quickly adapting to shifting conditions. Second, the collective observation needs to be efficiently shared to deduce the spatial relationship between agents in practical scenarios. While there are some recent works on multi-agent navigation [13], [14], they assume that agents accurately know their ground-truth positions within the environment and accordingly allocate goals to multiple agents. When data is independently collected by agents, global policy needs to exchange essential geometric information to coordinate movements effectively while avoiding redundant exploration. However, the precise spatial relationship may not be available in real-world deployment, resulting in suboptimal outcomes.

We address the two challenges by sharing lightweight topological memory among multiple agents. Specifically, each agent generates a topological graph whose nodes represent spatial regions. Among nodes, the boundaries between the explored and unexplored regions are marked as frontiers, which are candidates for exploration goals. Note that each agent generates the topological graphs independently and that no spatial relationship between agents is provided. The graph nodes contain minimal image features from visual observations at the location. These are the only information propagated between agents, making the entire process lightweight and

Manuscript received: May, 28, 2024; Revised August, 20, 2024; Accepted September, 13, 2024.

This paper was recommended for publication by Editor M. Ani Hsieh upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. RS-2023-00208197) and Creative-Pioneering Researchers Program through Seoul National University.

[1] The authors are with the Department of Electrical and Computer Engineering and INMC, Seoul National University, South Korea. {eunsunlee, youngmin.kim} @snu.ac.kr

*Young Min Kim is the corresponding author of this work.

Digital Object Identifier (DOI): see the top of this page.

versatile for different agent configurations. We additionally find a relative pose between agents using the topological memory and further improve efficiency. Our suggested refinement method quickly finds landmarks between the keyframe nodes and optimizes the agents' pose in real-time.

Our exploration strategy utilizes the explicit frontier-based method, which builds on a feature-space abstraction ignorant of the low-level configurations. After aggregating information from the topological graphs of other agents, the policy of each agent selects the optimal goal position in a distributed fashion. The data from all agents collectively generates similarity scores of the nodes by comparing the similarity between visual information in individual nodes in all graphs. Then, the agents individually generate their local exploration maps from which they select the exploration goals as the frontiers with the lowest similarity score in their topological maps. As a result, distributed exploration can comprehensively cover a large-scale area without building a coherent global map.

In summary, the main contributions of our study are as follows: i) We introduce a multi-agent exploration method, an off-the-shelf, readily deployable and adaptable to various scene configurations, utilizing both the frontier-based method and similarity score map generated from topological graph memory. ii) We suggest a fast and real-time multi-agent optimization method, based on keypoint feature matching, for more efficient exploration performance and map alignment, iii) We evaluate our method across various configurations, including scenarios with a large number of agents and expansive environments, to demonstrate the enhanced productivity of our multi-agent exploration policy.

II. Related Works

A. Multi-agent Navigation

The goal of multi-agent navigation is to collaboratively plan paths and assign objectives for multiple agents in complex environments. Various approaches have been developed to address the need for coordination in achieving the agents' collective objective. Building on single-agent navigation, decentralized policies [15]–[17] allow each agent to make decisions using shared information from other agents, when available. In contrast, centralized strategies [18], [19] operate on a single decision maker to coordinate goals for multiple agents. Notable works [2], [20] in cooperative exploration and strategic path planning optimizes the efficiency of navigation through region partitioning or probabilistic modeling. Recent studies [13], [14], [21] also leveraged neural networks or reinforcement learning to improve agents' navigation performance in dynamic environments. Despite these efforts, challenges persist in achieving a robust multi-agent navigation policy due to unpredictable novel environments, varying agent dynamics, and the need for robust communication protocols between agents. Thus, our proposed multi-agent navigation policy seeks to address these challenges with a novel framework that enhances the efficiency, adaptability, and cooperative behavior among agents.

B. Map Memory For Robot Navigation

Navigational embodied agents employ different memory types to store observations from their trajectories. Many navigation agents [2], [22], [23] typically utilize occupancy grid maps designed for spatially grounded path planning. However, these maps face challenges in adapting to changes in dynamic environments, often necessitating updates across the entire grid. Additionally, the absence of semantic and contextual information can result in the misidentification of areas with similar structures. Most significantly, it is challenging for multiple agents to synchronize their states using solely the information from the grid maps. To enhance semantic understanding, some approaches leverage scene graphs [12], [24], which represent environments with semantic relationships between objects. This helps agents better grasp spatial context, allowing them to prioritize and guide navigation based on high-level, task-relevant guidance. However, scene graphs are computationally intensive and may not scale well in multi-agent scenarios due to their complexity.

Thus, we adopt the structure of topological map memory. Navigation agents using topological maps [25]–[27] benefit from a simpler structure than scene graphs, reducing computational overhead and memory use. This simplicity is crucial for handling multiple agents across a broad exploration area. Topological maps emphasize paths and waypoints, providing spatially grounded data through nodes and edges, thus incorporating the benefits of grid maps. In multi-agent scenarios, topological maps ease coordination by providing a consistent, interpretable representation of trajectories, facilitating direct data sharing among agents, and improving collaborative exploration. Moreover, studies in real-world robotics [28], [29] have highlighted the robustness and effectiveness of topological maps, demonstrating their practicality and reliability for embodied agents' navigation in unpredictable scenes. Topological maps offer a straightforward, scalable, and coordination-friendly solution for multi-agent navigation, combining the advantages of both occupancy grid maps and scene graphs while mitigating their respective drawbacks.

C. Pretrained Vision Models

Pretrained models have become crucial for enhancing navigation task performance, capitalizing on their ability to be trained on extensive dataset and to generalize robustly across diverse environments [27], [30], [31]. The pretrained vision models, trained on a large image dataset, provide a broad understanding of visual features essential for robot navigation tasks [11], [22], [32], [33]. From raw RGB images, they analyze and interpret a complex hierarchy of visual inputs-from basic edges and textures at lower layers to more intricate elements like object types and layouts at higher layers. Additionally, these models increase the robustness of agents to variations such as changes in lighting and differences in input configurations like camera aspect ratios. Recent studies [29], [34], [35] have also leveraged vision-language models like CLIP [36], significantly advancing the integration of computer vision and natural language processing. These models enable agents to understand visual content in context with natural

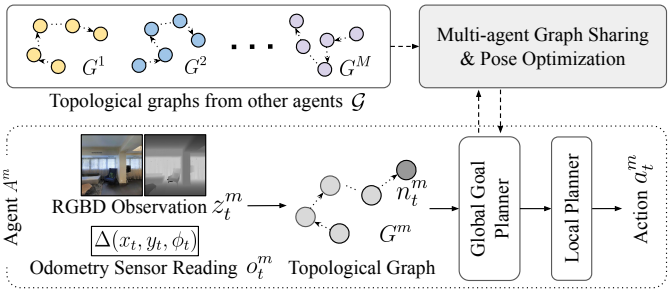


Fig. 1. **Overview of our approach** In our proposed strategy for multi-agent exploration, each agent A^m constructs a topological graph G^m based on RGBD observations z_t^m and odometry sensor data o_t^m collected at every timestep. The graphs from all agents, $\mathcal{G}$, are shared across agents and the relative poses between agents are optimized. Utilizing the collective graph information, the global goal planner selects a long-term goal. Subsequently, the local planner generates an action a_t^m for each step.

language cues, thus interpreting instructions by humans effectively. Employing these pretrained vision models not only generalizes robots to various environments but also diversifies their navigation tasks. Trained on vast datasets, they excel in adapting to new and unforeseen environments, demonstrating exceptional generalization and zero-shot transfer capabilities in robot navigation. Our representation can adapt diverse pretrained modules for low-level policy for individual agents while sharing minimal information between agents. In our proposed policy, we incorporate a pretrained visual encoder [37] trained on ImageNet dataset [38] to ensure the policy's generalization across various complex environments.

III. Method

We propose a novel and efficient exploration policy for controlling multiple agents $\{A^1, A^2, \ldots, A^M\}$ that are deployed in a single environment. The main objective is to propose a distributed policy so that the agents can quickly maximize the overall coverage and collaboratively build the map. Figure 1 outlines the overall navigation pipeline, with each iteration corresponding to a timestep, consistent with the conventional representation used in previous navigation studies [3], [22], [28]. At each time step t, the agent A^m traverses the scene and collects an RGBD observation z_t^m and odometry sensor measurement o_t^m as input. The agent also creates a node n_t^m of a topological graph G^m. Our proposed graph structure is designed to share essential information across different agents in real-time. Note that agents individually store raw measurements and can generate detailed map geometry, which is not shared between agents.

From the set of topological graphs $\mathcal{G} = \{G^1, G^2, \ldots, G^M\}$, the global goal planner only observes an abstract layer with lightweight features and efficiently provides high-level guidance to the agent. This abstraction helps address the challenges of homogeneous environments by utilizing information in a more generalized form. Then, the subsequent local planner achieves the assigned global goal by generating the next step actions. Agents can flexibly adapt to their configurations to interface with different sensor modalities or robot action spaces. We also utilize the shared graphs to optimize relative poses for deployed agents, allowing them to quickly aggregate detailed information and build a coherent map during exploration.

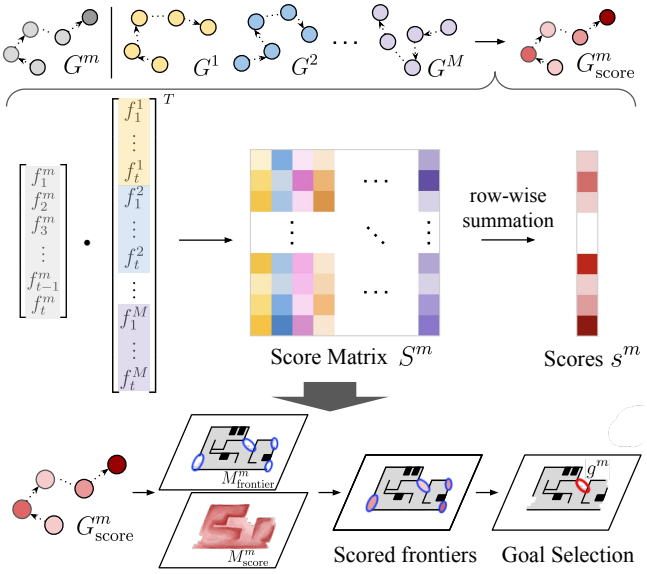


Fig. 2. **Global Goal Planner** Given the topological graphs $G^1, G^2, \ldots, G^M$ from other agents, $\mathcal{G}$, the agent A^m calculates score matrix S^m through the similarity features stored at each node. The summed scores per node are then updated on the topological graph to generate a scored graph, G^m_{score}. From the score graph, two layers of maps are generated, the frontier map M^m_{frontier} and the similarity score map M^m_{score}. Lastly, the frontier with the lowest similarity score is selected as the final global goal g^m.

Section III-A describes how the topological graph is generated for each agent. In Section III-B, we elaborate on the details of a global planner, which brings in several crucial aspects of our suggested multi-agent navigation strategy. Lastly, Section III-C explains our multi-agent pose optimization method.

A. Topological Map Memory Generation

The topological map memory stores the necessary information as the agent travels along the trajectory, making it easy to extract essential information to share among agents. The node $n_t \in N$ on the hierarchical graph $G = (N, E)$ stores information at each time step t. Each node n_t contains the agent's RGBD observation z_t, and the edge $e_{t,t+1} \in E$ stores the relative pose $(dx, dy, d\phi)$ between n_t and n_{t+1}, initialized by the odometry reading o_t. We use superscripts to denote indices to refer to the quantities of agents A^m when necessary.

This is in contrast to many previous studies that share an aggregated high-resolution map for all navigating agents [14], [20]. Our topological graph is a superior alternative, especially in noisy multi-agent navigation scenarios, because we can quickly adjust the pose errors and dynamically update the map. At any time step, the agent can combine the depth measurements with the relative pose data to generate an agent-specific map.

The nodes contain two additional types of features calculated from the image, which are selectively shared between multiple agents. First, we obtain similarity features f_t of the image extracted from a pre-trained encoder [37]. The encoder converts an image to a vector, and we can obtain pairwise similarity between two images by taking the inner product between the feature vectors. Secondly, we extract a sparse set of image feature points $\{x_{i,t}\}$, which are designed to find relative poses between images. The selected distinctive points

are often sufficient to match and refine relative poses between images, even those from different agents. Both features are calculated on the fly and stored in the nodes. The features are shared with other agents for the global goal planner or pose optimization in the overall exploration task.

We then use the similarity features f_t of nodes to select a subset of nodes $N_{\text{keyframe}} \subset N$, that can compactly represent the overall trajectory. Similar to [27], we select a node as the keyframe node only if the current node is uniquely different from the most recently added keyframe node. As the number of nodes increases, we use keyframe nodes to process further steps in a hierarchical manner.

Algorithm 1 Global Goal Planner

Input: Topological graphs, $\mathcal{G} = \{G^1, G^2, ..., G^M\}$
Output: Global goals for all agents, $g^1 \dots g^M$

1: **function** GETSCOREGRAPH($G^m, \{G^1 \dots G^M\}$)
2: $t \leftarrow$ length of G^m
3: $f_i^m \leftarrow$ feature from node n_i of graph G^m
4: $f_j^k \leftarrow$ feature from node n_j of graph G^k
5: scores $s^m = \begin{bmatrix} \sum_{k=1}^{M} \sum_{j=1}^{t} f_1^m f_j^k \\ \sum_{k=1}^{M} \sum_{j=1}^{t} f_2^m f_j^k \\ \vdots \\ \sum_{k=1}^{M} \sum_{j=1}^{t} f_t^m f_j^k \end{bmatrix}$
6: $G^m_{\text{score}} \leftarrow$ update the nodes of G^m with scores s^m
7: **return** G^m_{score}
8: **function** GENERATEMAPS(G_{score})
9: $N, E \in G_{\text{score}}$
10: **for** $i \leftarrow 1$ to len(N) **do**
11: $M_{\text{frontier},i} \leftarrow$ project map from (z_i, o_i)
12: $M_{\text{score},i} \leftarrow$ project map from (z_i, o_i, s_i)
13: $M_{\text{frontier}} \mathrel{+}= M_{\text{frontier},i}$, $M_{\text{score}} \mathrel{+}= M_{\text{score},i}$
14: **return** $M_{\text{score}}, M_{\text{frontier}}$
15: **procedure** GLOBALGOAL($\mathcal{G}$)
16: $M \leftarrow$ no. of agents
17: **for** $m \leftarrow 1$ to M **do**
18: G^m_{score} = GETSCOREGRAPH($G^m, \{G^1 \dots G^M\}$)
19: $M_{\text{score}}, M_{\text{frontier}}$ = GENERATEMAPS(G^m_{score})
20: global goal $g^m \leftarrow$ frontier with the lowest score
21: **return** global goals, $g^1 \dots g^M$

B. Global Goal Planner

Our global planner introduces a multi-agent exploration policy that can collaboratively enhance the likelihood of exhaustive exploration without training. In contrast to the learning-based policies for a single, isolated agent in a fixed set-up, we integrate the map with the traditional frontier-based method to determine the explicit exploration goals for individual agents. Our global planner distributes the goals among agents whose spatial relationships are unknown by estimating the frequency of visits within the mapped area. In high-level, the global planner follows three steps as described in Algorithm 1 and Figure 2. Individual agents first mark similarity scores for their observation nodes and generate score graph G^m_{score}. Then each agent generates two specific maps: an occupancy map M^m_{frontier} and a similarity score map M^m_{score}. Then the planner determines the global goal location g^m by examining the two maps. Triggered every 50 timesteps, our global goal planner uses sparse communication strategy to ensure robust performance, even with intermittent or unstable connections in a multi-agent scenario.

In the beginning of each global goal planning process, agents share their topological maps. We then augment each agent node N^m in G^m with similarity scores, which measure the affinity between each agent's visual feature descriptors f_i^k against the remaining agent's observations (lines 1-7). In principle, the score value corresponding to a node is high if other graphs contain similar observations in terms of the similarity features. The feature similarities are first collected in a score matrix $S^m \in \mathbb{R}^{t \times (M-1)t}$ where each row indicates a node of the current agent of interest $n_i^m \in N^m$, and each column represents the comparison against nodes from $(M-1)$ other agents (line 5). The entries of S^m are similarities between corresponding nodes calculated as pair-wise dot products of the similarity feature f_i^k. The similarity score of all nodes $s^m \in \mathbb{R}^t$ is then estimated as row-wise sums of S^m. In Algorithm 1, the function `GetScoreGraph` describes how the summed score encapsulates the node's feature similarity across all considered agents. Here, f_i^m denotes the feature vector of node i in G^m, and f_j^k denotes the feature vector of node j in graph G^k of agent A^k (lines 3-4).

The resulting scores s^m are then updated to each corresponding node to generate the agent's scored topological graph, denoted as G^m_{score} (line 6). This method efficiently accounts for the feature similarity among nodes from different agents' topological graphs, providing a robust means to share strategic information for multi-agent navigation and exploration.

With the scored topological graph G^m_{score}, we build two distinct map layers, namely free-space map and similarity score map, as described in function `GenerateMaps` (lines 8-14). As positional information is unknown, each agent assumes its start position to be the origin of the maps. The first layer, the free-space map M^m_{frontier}, is constructed by projecting egocentric observations onto their observed positions. We then find regions at the interface of open and unexplored space (line 11). These frontier regions serve as potential global goal candidates for further exploration by the global planner. The second layer, the similarity score map M^m_{score}, involves overlaying the similarity score across the explored areas based on the positions stored in the graph's nodes (line 12). Nodes within the score graph G^m_{score} receive higher similarity scores when their observations align closely with those of other agents, thus highlighting areas that are frequently visited within the scene.

Using the two agent-centric maps, our policy selects the global goal g^m to be the one with a low similarity score among goal candidates of frontiers (line 20). The similarity score map integrates data from all agents and indicates the less-explored areas. This geometric approach enables the efficient performance of multiple agents in any test scene, especially in large-scale scenarios, presenting a significant benefit over

TABLE I
EXPLORATION RESULT EXPLORATION RATIO (%) OF MULTIPLE AGENTS DEPLOYMENT IN ENVIRONMENTS WITH NOISY ACTUATION.

	Scene Size	Small Scenes ($< 75m^2$)				Medium Scenes ($75 - 250m^2$)				Large Scenes ($> 250m^2$)			
	No. Agents	1	2	3	5	1	2	3	5	1	2	3	5
(a) Agents without Odometry Noise	Greedy	90.67	92.12	94.98	94.86	61.06	66.29	71.25	75.34	33.04	35.87	42.51	48.32
	ANS	85.90	90.75	92.19	92.94	49.05	57.38	60.07	61.92	24.70	35.01	36.10	40.82
	MAScan	88.07	94.84	95.45	96.34	**70.00**	**81.65**	**84.87**	86.13	37.85	45.94	51.27	54.84
	Ours	**93.08**	91.13	95.77	96.31	68.49	76.46	80.99	87.86	**37.49**	49.06	57.23	66.96
	Ours+Opt	**93.08**	**95.81**	**96.10**	**96.68**	68.49	78.62	83.10	**88.91**	**37.49**	**52.04**	**61.26**	**70.25**
(b) Agents with Identical Odometry Noise	Greedy	87.02	92.58	**95.94**	96.04	53.94	65.19	71.86	75.12	32.93	35.98	41.17	50.16
	ANS	86.49	91.20	91.80	92.82	45.93	58.08	60.42	62.80	26.27	32.86	37.30	39.55
	MAScan	**92.81**	95.03	95.47	95.97	**66.94**	77.29	83.16	86.13	**31.37**	43.21	51.43	54.76
	Ours	90.44	93.92	95.51	96.40	64.04	75.15	80.95	85.79	30.75	44.32	53.04	61.65
	Ours+Opt	90.44	**95.38**	95.52	**96.64**	64.04	**78.42**	**83.54**	**89.09**	30.75	**49.27**	**55.63**	**65.09**
(c) Agent with Varying Odometry Noises	Greedy	87.08	94.35	94.86	95.47	56.74	68.87	71.29	75.49	31.04	37.75	45.17	51.41
	ANS	86.31	91.27	91.90	92.82	47.78	56.62	59.01	61.63	26.08	35.04	35.71	40.13
	MAScan	91.58	**94.84**	95.20	96.48	**63.29**	**77.02**	**84.49**	86.57	**32.95**	45.86	49.19	55.85
	Ours	**92.56**	89.51	94.56	94.99	61.39	74.07	80.35	82.50	31.80	43.43	46.59	60.11
	Ours+Opt	**92.56**	92.91	**95.22**	**96.52**	61.39	72.87	80.34	**87.49**	31.80	**50.04**	**58.27**	**63.19**

learning-based methods that are confined to specific training configurations.

C. Multi-agent Pose Optimization

While the similarity features from nodes are sufficient to bias agents towards unexplored regions, we cannot build a coherent global map after exploration without accurate spatial relationships between them. We include a multi-agent pose optimization method within the pipeline based on the keypoint match algorithm [39]. Our method intermittently assesses the degree of overlap between the agent's explored areas. Given a pair of RGBD images capturing the overlapping area, we can determine the cameras' relative positions by matching keypoints $\{x_{i,t}^k\}$. We find the matches across frames from different agents and designate them as landmarks between the agents' topological graphs, building a pose graph with refinements. We further remove outlier matches based on the projected distance. Then we can optimize the pose graphs of multiple agents, utilizing the Levenberg-Marquardt algorithm from the GTSAM solver [40]. We trigger optimization when the number of established landmarks exceeds a predefined threshold. Successful optimization updates the pose data within topological graphs, ensuring that the agents' positions are accurately aligned. We incorporate a hierarchical structure and search for landmarks only for the keyframe nodes of the topological graph when real-time performance is required.

With this online pose optimization strategy, our global policy can also integrate these updated relational poses into its operational framework. Our policy builds a comprehensive similarity score map applicable to all agents. The frontier map and goal candidates are generated in an agent-centric manner as before, with the final goal selected based on the comprehensive similarity score map. After successful optimization, all agents operate cohesively in their shared environment.

IV. EXPERIMENTS

We demonstrate the advantages of our multi-agent exploration policy across various experimental setups. We assess its performance by deploying multiple agents in a single novel scene. Each agent operates simultaneously within the same environment, allowing us to measure the total area explored by all agents. We report the exploration ratio, following [1], after allowing the agents to traverse for a fixed number of steps in diverse scene scales with different numbers of agents. It is important to note that the agents are not provided any ground-truth spatial relationship between them, relying solely on their egocentric observations for navigation. We employ the LoCobot model [41] as our robot platform. In simulator, we configure the robot to execute each discrete action with a linear speed of 0.75m/s and an angular speed of 0.9 rad/s. Our agent is built on Pytorch [42] and optimized for performance with an RTX 2080 GPU.

A. Baselines

We conduct thorough comparisons of our policy against a range of state-of-the-art baselines. To ensure a fair comparison, we only replace the global goal planner with the respective baselines. The "**Greedy**" model represents the most recognized traditional frontier-based method which selects the goal nearest to the agent based on geodesic distance. The "**ANS**" model, introduced by Chaplot et al. [22], employs a reinforcement learning strategy for single-agent exploration by directly regressing the global goal from each agent's occupancy grid map. Both baselines embody fully decentralized multi-agent policies, allowing robots to set their objectives based on maps they construct independently, without any inter-agent communication. We also evaluate the performance of the state-of-the-art multi-agent exploration policy, "**MAScan**" [20]. In the original paper, robots are co-located initially to take 360°scans for calibrating their relative poses. In our implementation, we simply provide the pose information. Thus, this baseline is a centralized global planner that operates under a strong assumption that the ground-truth poses of multiple agents are known. It generates k-means clusters over all frontiers found from the combined explored map from all agents and assigns a cluster to each agent based on optimal mass transport formulation.

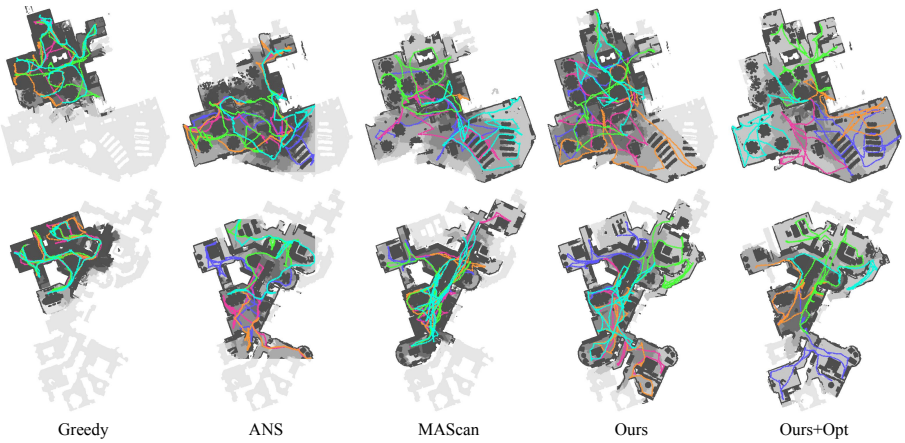


Fig. 3. **Visual results of multi-agent exploration** Our suggested policy compared with state-of-the-art approaches on two scene samples from the Matterport3D dataset. The reconstructed maps (gray) and each of the five agents' trajectories are visualized with different colors.

Each agent then selects a frontier from the assigned cluster by solving a vanilla traveling salesman problem. In our main experiment, we present our proposed policy as "**Ours**" and further assess it by incorporating the suggested multi-agent pose optimization strategy, which we refer to as "**Ours+Opt**".

B. Main Experiment

Our off-the-shelf exploration agent is tested on Matterport3D dataset [43] using the Habitat simulator [44]. We reported the average results from 61 experiments conducted across 61 different scenes and categorize the results by three scene sizes: small ($< 75m^2$), medium ($75m^2$-$250m^2$), and large ($> 250m^2$). Each agent explores the scene for 1,000 steps with the actuation noise following the model from [45]. For evaluation purposes, all agents start from the same location, though each agent is designated a unique initial pose, and remain unaware of each other's pose. Table I-(a) illustrates that our policy consistently outperforms other baseline approaches. Its effectiveness is further enhanced when integrated with the multi-agent optimization strategy (Ours+Opt). Despite MAScan incorporating the ground-truth spatial relationship between agents, our policy outperforms it in the majority of experiments, or demonstrates comparable performances in some medium-sized scenes. This result highlights the effectiveness of the similarity score map in our proposed method for coordinating global goals for multiple agents. Furthermore, our result indicates that the performance gap widens favorably as scene size increases and more agents are deployed. Therefore, these findings emphasize the superior efficiency of our exploration strategy, particularly in large-scale scenes with numerous agents, compared to other state-of-the-art baselines and demonstrate its advantage over single-agent policies.

We also present the qualitative result of our multi-agent exploration in Figure 3. It illustrates the exploration outcome for five deployed agents across two large-sized scenes, with each row displaying results from the same scene and each column representing different baselines. Coverage frequency by varying numbers of agents is depicted using different shades of gray, where the paths of individual agents are marked with distinctively colored lines. As shown, our policy achieves the most comprehensive and evenly distributed area coverage compared to the other baselines. Agents using Greedy policy often explore overlapping areas, whereas our Ours+Opt model significantly reduces such overlap, enhancing overall exploration efficiency among multiple agents. In conclusion, the visual result aligns with the quantitative findings from Table I-(a), showing a consistent trend in our policy's performance.

Furthermore, we assess our policy in a scenario where the agents encounter both actuation and odometry sensor reading noise. We implement a pretrained pose estimator designed from [22] across all evaluated policies to minimize sensor pose errors, yet precise poses remain unobtainable, leading to cumulative trajectory errors. Such inaccuracies can degrade map quality and negatively impact the performance of frontier selection pipelines or map-based planning methods. Thus, we evaluate our policy and other baselines under these noisy conditions, as detailed in Table I-(b). Interestingly, ANS, a single-agent policy trained in a similar noisy setting, exhibits the smallest performance decline in small-sized scenes with a single agent deployed. Nevertheless, our policy consistently maintains high-performance levels compared to other baselines in most experimental cases.

Lastly, to evaluate our method's performance under unreliable and varied odometry conditions, we conducted an additional experiment where the agents operate with different levels of odometry sensor noise, as shown in Table I-(c). For instance, in a scenario with three agents, we tested one agent with noiseless odometry and the other two with different levels of odometry noise. The results demonstrate that our proposed policy outperforms other state-of-the-art methods under these challenging conditions with a more number of agents in large scenes, underscoring its stable scalability in the multi-agent set-up. It is important to note that in our main experiments, MAScan uses ground-truth relative poses between agents, which may give it an inherent advantage in performance under the experimental conditions. In conclusion, across all three experimental setups, our policy shows a growing performance advantage as scene size gets larger and more agents are deployed, even amidst noisy conditions.

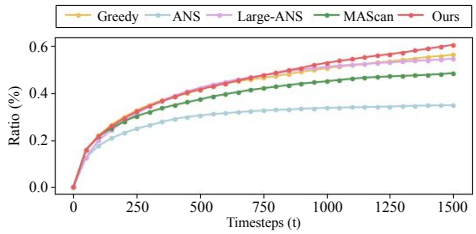


Fig. 4. **Result of Large Scale Scene Experiment** The exploration ratio of 10 agents deployed in Stanford 2D-3D scenes over 1500 steps.

C. Large Scale Scene Experiment

To evaluate the effectiveness of our policy in expansive environments and with a large number of agents, we also evaluate its performance using the 6 scenes from Stanford 2D-3D dataset [46] on the iGibson simulator [47]. We report the average results from 20 experiments, with agents starting from different positions in each episode within the same scene. While the average scene size in the Matterport3D dataset is $517m^2$, the sizes of the Stanford 2D-3D scenes are considerably larger, ranging from $450m^2$ to $1700m^2$. In Figure 4, we showcase the exploration result of 10 agents over 1,500 steps. Our policy significantly outperforms the baselines in terms of exploration ratio. Moreover, its performance advantage becomes more pronounced as the agents continue to explore over a longer duration. In this experiment, the map sizes are beyond ANS could handle, and its performance significantly deteriorates. We introduce an additional baseline named "**Large-ANS**," which we expanded the map memory that is generously sized to fit the test scene. We then dynamically re-center and crop the map from the expanded memory to plan goals from the same input dimensions outlined in the ANS model. This approach allows the model to cover a larger area while still adhering to the pre-defined configurations of the ANS model. Although the performance of Large-ANS baseline significantly improves over ANS, it still falls short of surpassing our policy. An additional noteworthy observation is that the Greedy baseline performs comparatively better in this setting than in previous experiments. We attribute this improvement to the structural characteristics of the Stanford 2D-3D dataset, where many rooms are aligned along extensive hallways. Figure 5 illustrates that the area explored by agents using our policy is far larger and more evenly distributed than that covered by other baselines. These experiments conclusively demonstrate that our policy delivers exceptional efficiency in large scale scenes with a significant number of agents.

D. Computational Efficiency Assessment

Our policy employs a light-weight topological graph that uses 2.66MB of RAM for 1,000 nodes, which is significantly less than the 9.66 MB required by the ANS model's global planner. Each additional 100 nodes only adds 0.26MB to memory usage. Additionally, our global planner operates with an average runtime of 2.71 seconds and does not require prior training, allowing for immediate and straightforward deployment and providing a distinct advantage over learning-based models. With a per-timestep runtime of approximately 0.36 seconds, our policy is well-suited for real-time operation.

V. Conclusion

To summarize, we developed a multi-agent exploration policy that provides an efficient communication method among agents and strategically assigns goals to optimize their collective objectives. Our proposed method is particularly effective and well-suited for expansive environments with a large number of agents, where its capability to coordinate and optimize agent actions is highly advantageous.

References

[1] Devendra Singh Chaplot, Ruslan Salakhutdinov, Abhinav Gupta, and Saurabh Gupta. Neural topological slam for visual navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12875–12884, 2020.

[2] Santhosh K Ramakrishnan, Ziad Al-Halah, and Kristen Grauman. Occupancy anticipation for efficient exploration and navigation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, pages 400–418. Springer, 2020.

[3] Saurabh Gupta, James Davidson, Sergey Levine, Rahul Sukthankar, and Jitendra Malik. Cognitive mapping and planning for visual navigation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2616–2625, 2017.

[4] Devendra Singh Chaplot, Murtaza Dalal, Saurabh Gupta, Jitendra Malik, and Russ R Salakhutdinov. Seal: Self-supervised embodied active learning using exploration and 3d consistency. *Advances in Neural Information Processing Systems*, 34, 2021.

[5] Christian Dornhege and Alexander Kleiner. A frontier-void-based approach for autonomous exploration in 3d. *Advanced Robotics*, 27(6):459–468, 2013.

[6] Ana Batinovic, Tamara Petrovic, Antun Ivanovic, Frano Petric, and Stjepan Bogdan. A multi-resolution frontier-based planner for autonomous 3d exploration. *IEEE Robotics and Automation Letters*, 6(3):4528–4535, 2021.

[7] Cheng Zhu, Rong Ding, Mengxiang Lin, and Yuanyuan Wu. A 3d frontier-based exploration tool for mavs. In *2015 IEEE 27th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 348–352. IEEE, 2015.

[8] Brian Yamauchi. A frontier-based approach for autonomous exploration. In *Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97.'Towards New Computational Principles for Robotics and Automation'*, pages 146–151. IEEE, 1997.

[9] Haoran Li, Qichao Zhang, and Dongbin Zhao. Deep reinforcement learning-based automatic exploration for navigation in unknown environment. *IEEE transactions on neural networks and learning systems*, 31(6):2064–2076, 2019.

[10] Anna Guerra, Francesco Guidi, Davide Dardari, and Petar M Djurić. Reinforcement learning for uav autonomous navigation, mapping and target detection. In *2020 IEEE/ION Position, Location and Navigation Symposium (PLANS)*, pages 1004–1013. IEEE, 2020.

[11] Devendra Singh Chaplot, Dhiraj Prakashchand Gandhi, Abhinav Gupta, and Russ R Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. *Advances in Neural Information Processing Systems*, 33, 2020.

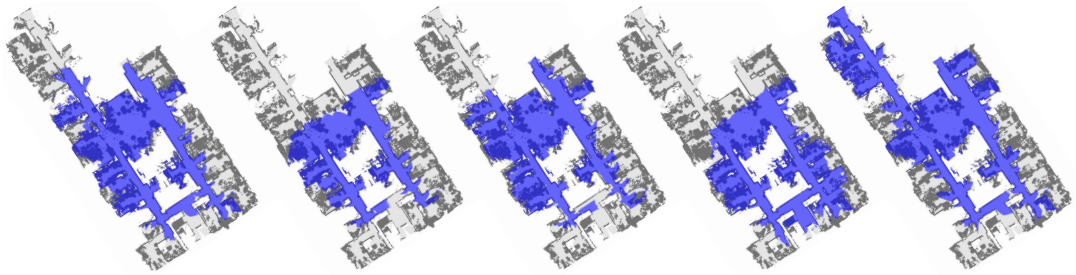
[12] Xinzhu Liu, Di Guo, Huaping Liu, and Fuchun Sun. Multi-agent embodied visual semantic navigation with scene prior knowledge. *IEEE Robotics and Automation Letters*, 7(2):3154–3161, 2022.

[13] Xinyi Yang, Yuxiang Yang, Chao Yu, Jiayu Chen, Jingchen Yu, Haibing Ren, Huazhong Yang, and Yu Wang. Active neural topological mapping for multi-agent exploration. *IEEE Robotics and Automation Letters*, 9(1):303–310, 2023.

[14] Kai Ye, Siyan Dong, Qingnan Fan, He Wang, Li Yi, Fei Xia, Jue Wang, and Baoquan Chen. Multi-robot active mapping via neural bipartite graph matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14839–14848, 2022.

[15] Pei Xu and Ioannis Karamouzas. Human-inspired multi-agent navigation using knowledge distillation. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8105–8112. IEEE, 2021.

[16] Roya Firoozi, Laura Ferranti, Xiaojing Zhang, Sebastian Nejadnik, and Francesco Borrelli. A distributed multi-robot coordination algorithm for navigation in tight environments. *arXiv preprint arXiv:2006.11492*, 2020.



Greedy ANS Large-ANS MAScan Ours

Fig. 5. **Visual result of large scale scene experiment** Our policy is compared to the state-of-the-art multi-agent exploration policies in a scene from the Stanford 2D-3D dataset.

[17] Rubisson Duarte Lamperti and Lucia Valéria Ramos de Arruda. Distributed strategy for communication between multiple robots during formation navigation task. *Robotics and Autonomous Systems*, 169:104509, 2023.

[18] Fethi Matoui, Boumedyen Boussaid, Brahim Metoui, and Mohamed Naceur Abdelkrim. Contribution to the path planning of a multi-robot system: centralized architecture. *Intelligent Service Robotics*, 13(1):147–158, 2020.

[19] Jingjin Yu and Steven M LaValle. Optimal multirobot path planning on graphs: Complete algorithms and effective heuristics. *IEEE Transactions on Robotics*, 32(5):1163–1177, 2016.

[20] Siyan Dong, Kai Xu, Qiang Zhou, Andrea Tagliasacchi, Shiqing Xin, Matthias Nießner, and Baoquan Chen. Multi-robot collaborative dense scene reconstruction. *ACM Transactions on Graphics (TOG)*, 38(4):1–16, 2019.

[21] Ruihua Han, Shengduo Chen, and Qi Hao. Cooperative multi-robot navigation in dynamic environment with deep reinforcement learning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 448–454. IEEE, 2020.

[22] Devendra Singh Chaplot, Dhiraj Gandhi, Saurabh Gupta, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using active neural slam. *arXiv preprint arXiv:2004.05155*, 2020.

[23] Eun Sun Lee, Junho Kim, SangWon Park, and Young Min Kim. Moda: Map style transfer for self-supervised domain adaptation of embodied agents. In *European Conference on Computer Vision*, pages 338–354. Springer, 2022.

[24] Johanna Wald, Helisa Dhamo, Nassir Navab, and Federico Tombari. Learning 3d semantic scene graphs from 3d indoor reconstructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3961–3970, 2020.

[25] Nikolay Savinov, Alexey Dosovitskiy, and Vladlen Koltun. Semi-parametric topological memory for navigation. *arXiv preprint arXiv:1803.00653*, 2018.

[26] Meera Hahn, Devendra Singh Chaplot, Shubham Tulsiani, Mustafa Mukadam, James M Rehg, and Abhinav Gupta. No rl, no simulation: Learning to navigate without navigating. *Advances in Neural Information Processing Systems*, 34:26661–26673, 2021.

[27] Nuri Kim, Obin Kwon, Hwiyeon Yoo, Yunho Choi, Jeongho Park, and Songhwai Oh. Topological semantic graph memory for image-goal navigation. In *Conference on Robot Learning*, pages 393–402. PMLR, 2023.

[28] Dhruv Shah, Benjamin Eysenbach, Gregory Kahn, Nicholas Rhinehart, and Sergey Levine. Ving: Learning open-world navigation with visual goals. *arXiv preprint arXiv:2012.09812*, 2020.

[29] Dhruv Shah, Błażej Osiński, Sergey Levine, et al. Lm-nav: Robotic navigation with large pre-trained models of language, vision, and action. In *Conference on robot learning*, pages 492–504. PMLR, 2023.

[30] Dhruv Shah, Ajay Sridhar, Nitish Dashora, Kyle Stachowicz, Kevin Black, Noriaki Hirose, and Sergey Levine. Vint: A foundation model for visual navigation. *arXiv preprint arXiv:2306.14846*, 2023.

[31] Chenguang Huang, Oier Mees, Andy Zeng, and Wolfram Burgard. Visual language maps for robot navigation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10608–10615. IEEE, 2023.

[32] Devendra Singh Chaplot, Helen Jiang, Saurabh Gupta, and Abhinav Gupta. Semantic curiosity for active visual learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VI 16*, pages 309–326. Springer, 2020.

[33] Theophile Gervet, Soumith Chintala, Dhruv Batra, Jitendra Malik, and Devendra Singh Chaplot. Navigating to objects in the real world. *Science Robotics*, 8(79):eadf6991, 2023.

[34] Vishnu Sashank Dorbala, Gunnar Sigurdsson, Robinson Piramuthu, Jesse Thomason, and Gaurav S Sukhatme. Clip-nav: Using clip for zero-shot vision-and-language navigation. *arXiv preprint arXiv:2211.16649*, 2022.

[35] Arjun Majumdar, Gunjan Aggarwal, Bhavika Devnani, Judy Hoffman, and Dhruv Batra. Zson: Zero-shot object-goal navigation using multimodal goal embeddings. *Advances in Neural Information Processing Systems*, 35:32340–32352, 2022.

[36] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[37] Junnan Li, Pan Zhou, Caiming Xiong, and Steven CH Hoi. Prototypical contrastive learning of unsupervised representations. *arXiv preprint arXiv:2005.04966*, 2020.

[38] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[39] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4938–4947, 2020.

[40] Frank Dellaert. Factor graphs and gtsam: A hands-on introduction. *Georgia Institute of Technology, Tech. Rep*, 2:4, 2012.

[41] *"LoCoBot: An open source low cost robot"*. http://www.locobot.org/, 2021.

[42] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32:8026–8037, 2019.

[43] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *arXiv preprint arXiv:1709.06158*, 2017.

[44] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A Platform for Embodied AI Research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[45] Adithyavairavan Murali, Tao Chen, Kalyan Vasudev Alwala, Dhiraj Gandhi, Lerrel Pinto, Saurabh Gupta, and Abhinav Gupta. Pyrobot: An open-source robotics framework for research and benchmarking. *arXiv preprint arXiv:1906.08236*, 2019.

[46] Iro Armeni, Sasha Sax, Amir R Zamir, and Silvio Savarese. Joint 2d-3d-semantic data for indoor scene understanding. *arXiv preprint arXiv:1702.01105*, 2017.

[47] Chengshu Li, Fei Xia, Roberto Martín-Martín, Michael Lingelbach, Sanjana Srivastava, Bokui Shen, Kent Elliott Vainio, Cem Gokmen, Gokul Dharan, Tanish Jain, Andrey Kurenkov, Karen Liu, Hyowon Gweon, Jiajun Wu, Li Fei-Fei, and Silvio Savarese. igibson 2.0: Object-centric simulation for robot learning of everyday household tasks. In Aleksandra Faust, David Hsu, and Gerhard Neumann, editors, *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 455–465. PMLR, 08–11 Nov 2022.