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

Learning Hierarchical Graph-based Policy for Goal-reaching in Unknown Environments

Yuxiang Cui, Shuhao Ye, Xuecheng Xu, Hao Sha, Cheng Wang, Longzhong Lin,
Yifei Yang, Jiyu Yu, Zhe Liu, Rong Xiong, and Yue Wang

***Abstract*— Goal-reaching in unknown environments is one of the essential tasks in robot applications. Large-scale perception and long-horizon decision-making are the keys to solving this task as the operation scope expands or complexity rises. Existing navigation methods may suffer from degraded performance in complicated environments induced by scalability-limited map representation or greedy decision strategy. We propose the path-extended graph as a compact map representation providing sufficient structural information within a reasonable receptive field and incorporate it into a hierarchical policy for higher efficiency and generalizability. The path-extended graph contains the concise topology of environment structure and frontier layout for large-scale perception, avoiding the impact of redundant information. The hierarchical policy solves long-horizon non-myopic decision-making through a high-level frontier selection policy using deep reinforcement learning (DRL) and a low-level motion controller that handles path planning and collision avoidance. Simulation and real-world experiments demonstrate that our method outperforms other competitive approaches in avoiding redundant movement and achieves efficient goal-reaching, especially in complex environments.**

I. Introduction

Autonomous robots capable of navigating unknown environments efficiently hold promise in bringing practical assistance to various applications like emergency rescue and target search. When carrying out such missions, the robots need to reach goals in large-scale, unmapped areas, which may be located at considerable distances. Thoroughly understanding and managing the complexities of these expansive and intricate environments is critical to avoid unnecessary movements and local optima challenges.

In classic methods, the feasibility of navigation in unknown environments is often underpinned by continuous map updating and incremental frontier-oriented planning. Despite the effectiveness, resource-consuming dense map construction and greedy frontier selection policy may lead to inefficient performance in large-scale environments and long-horizon tasks, impeding the flexible applications of these methods [1], [2]. Learning-based methods like Deep Reinforcement Learning (DRL) using Convolutional Neural Networks (CNN) [3], [4] have been proven to be able to mitigate the myopic decision problems by considering long-term returns with map information, but expensive mapping is still highly relied upon.

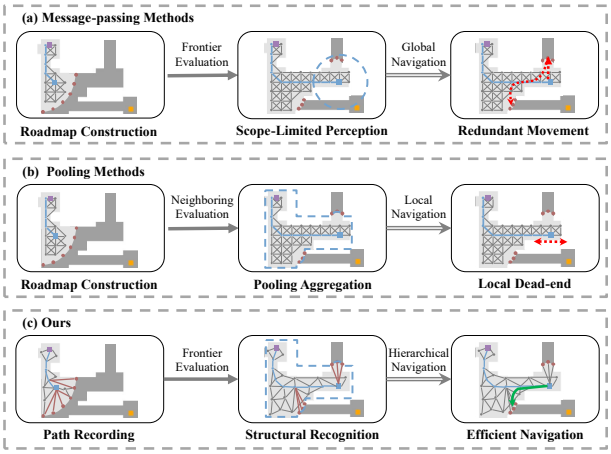


Fig. 1. Illustration comparing (a) message passing methods, (b) pooling methods, and (c) our method. Note that our method utilizes the path-extended graph as a compact map representation and integrates it into a hierarchical framework for higher scalability and stability.

Topological representations can provide promising solutions to the resource-consuming problems of their map-based counterparts, which have been proven effective in navigation tasks [5]–[9]. The graph representation characterizes the relations between explored entities with sparse structural information instead of over-provisioning dense information. Recent advances in topology-aware decision-making have leveraged Graph Neural Networks (GNN) with message-passing mechanisms and Transformer networks with attention mechanisms. These methods demonstrate stronger capabilities in encoding diverse topology and offer the advantage of consistent computation time, circumventing the escalating computational demands of extensive node-level simulation and evaluation inherent in navigation processes [10]. However, the perception capability of these two methods may degrade markedly in long-horizon tasks due to the over-smoothing problem in GNN and high resource occupancy in Transformer. Redundant graph expansion greatly hinders the compatibility between the graph and the networks with limited receptive fields, especially in complex environments that demand large-scale perception.

Some approaches adopt pooling methods to enlarge the receptive field by extracting global features instead of conducting local information aggregation. Directly mapping the features to local-scope controls like discrete actions or neighboring positions can be effective in some exploration tasks [11]–[13]. However, the sample efficiency of these methods may decrease significantly when deployed in long-horizon tasks due to sparse rewards and long episodes, making them

This work was supported in part by the National Nature Science Foundation of China under Grant 62373322 and in part by the Zhejiang Provincial Natural Science Foundation of China (LD22E050007).

The authors are with the State Key Laboratory of Industrial Control Technology and Institute of Cyber-Systems and Control, Zhejiang University, Zhejiang, China. Yue Wang is the corresponding author(e-mail: yuxiangcui@zju.edu.cn, 12332093@zju.edu.cn, xuechengxu@zju.edu.cn, shahao@zju.edu.cn, chengwang@zju.edu.cn, linlongzhong2000@zju.edu.cn, lzsunny725910@126.com, rxiong@zju.edu.cn, wangyue@iipc.zju.edu.cn).

hard to train and vulnerable to large-scale environments [14].

In this paper, we present the path-extended graph for concise map representation, facilitating large-scale perception, and we develop a hierarchical policy designed for efficient long-horizon decision-making in goal-reaching tasks within unknown and complex environments. The path-extended graph describes the global structure and frontier layout for the explored regions, enabling structural recognition for neural networks under a limited receptive field. The hierarchical policy directs the DRL-based policy to focus on non-myopic frontier selection in the global scope and leaves the local navigation problems to robust motion planners, taking the complementary strength of both DRL-based policy and mature navigation algorithms. The frontier selection policy breaks down the long-horizon goal-reaching problem into sequential waypoint-approaching tasks and the motion controller handles path planning and dynamic collision avoidance between the waypoints without global dense mapping.

In summary, the main contributions of this work include:

- We propose the path-extended graph as a map representation for unknown environments with concise information describing both global structure and frontier layout, enabling effective large-scale perception.
- We propose a hierarchical architecture composed of a high-level DRL-based frontier policy and a low-level controller, efficiently solving long-horizon planning.
- The system is validated in various simulated and real-world settings, showing its efficiency and scalability in goal-reaching while minimizing redundant movements and avoiding dead-ends.

II. Related Works

A. Map Representation

Classic goal-reaching navigation systems highly depend on the pre-constructed dense maps for precise path planning. Dense metric maps have to be obtained with the help of human control, predefined instructions, or intelligent policies before deployment, and the robots can only complete goal-reaching within the mapped regions [4], [15]–[17]. But as exploration for mapping methods, they mainly focus on mapping the whole environment instead of reaching the goal position with only necessary exploration or coarse environmental information [12], [18]. To relieve expensive mapping stress, the topological map is an emerging lightweight representation containing sparse structural information of the environment layout [19]–[21]. Early works use sampling-based methods like Rapidly-exploring Random Trees (RRTs) and Probabilistic Roadmaps (PRMs) to build a sparse graph representation of the environment to reduce resource consumption and enhance generalization performance [14], [19], [22]. Nodes in recent graph-based methods are usually spatially sampled in open space [11] [23] or extracted from historical robot poses [7], and edges are constructed between them according to accessibility or human expert evaluation in belief space [8]. Plenty of works show that topological maps can be useful in describing high-level environmental information [5], [24], [25] in tasks under a relatively small number of nodes. However, as the graph expands with the exploration, the number of nodes may increase quickly, which brings difficulty to neural networks with limited sensing fields or transformer networks with high computation resource requirements. Methods like hierarchical hops sensing mechanisms [11] are proposed to enlarge the receptive field, but a lot of task-irrelevant nodes are still under consideration. The vacancies in enhancing the compatibility between the map representation and the policy from the graph construction perspective are worth to be filled.

B. Policy Architecture

Traditional methods split the navigation problems into two stages, global path planning and local trajectory planning. As for goal-reaching tasks, an additional sub-goal decision step is needed for guidance to direction with higher value, usually involving selecting positions of interest like frontiers and next-best viewpoints [26]. Early works tend to utilize greedy strategies ignoring the structural information of the environments that could be important to guide the robots to avoid problems like getting stuck in local optimum [27]. Learning-based methods have been proposed to mitigate myopic decision problems in navigation tasks recently, taking global environmental layout into consideration by using topological environmental information [23] or sequential image-based map information [3], [4]. Some researchers directly use end-to-end frameworks that map the observations to control commands like velocity controls [12], [28] or neighboring positions [11], which can achieve agile performance in various scenarios but may suffer from problems like difficult long-horizon planning learning or vulnerable transferring performance [3], [14]. Methods with modular pipelines are trending these days for their generalization stability in various simulation environments and real-world environments [3], [26], [29]. Some works combine planning in pre-mapped environments with learning-based collision avoidance policies for long-horizon deployment [14], [30]. Methods for image-goal navigation or object-goal navigation utilize modular pipelines to complete tasks in a coarse-to-fine style, in which coarse-grained direction selection considering the semantic similarity and topological layout guides fine exploitation in the local scope [31].

C. Goal-reaching in Unknown Environments

Different from exploration for mapping tasks, goal-reaching in unknown environments demands the robot to complete target-approaching in previously unknown environments. Cimurs et al. present a map-based strategy using points of interest considering both information gain and distance cost as navigation guidance for a local DRL-based collision avoidance policy [18]. Ravichandran et al. adopt an exploration policy with scene-graph-based representation for multi-object search tasks in which the purpose is to cover as many objects as possible, instead of arriving at a specific position [12]. There is a dearth of work that utilizes graphs as lightweight environmental memory for goal-reaching in unknown environment tasks, inspiring us to utilize a hierarchical goal-reaching policy architecture leveraging the path-extended graph as the map representation.

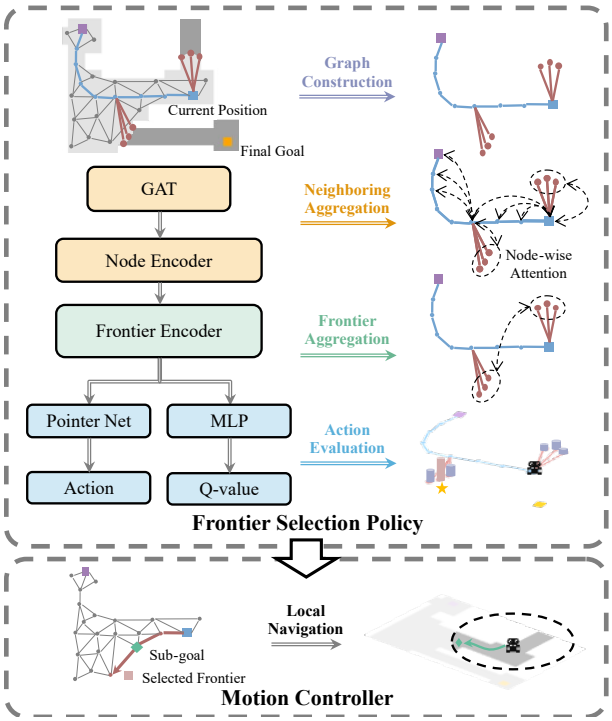


Fig. 2. Framework overview. We adopt a hierarchical pipeline composed of an event-triggered frontier selection policy and a time-triggered motion controller. The path-extended graph is extracted from the continuously updated roadmap and passed to policy networks. Node-wise attention mechanism (dashed arrows) aggregates information between neighboring nodes and frontiers for multi-scale perception. Through action evaluation, the frontier with the highest attention weight (yellow star) is selected as a waypoint for the motion controller, guiding sub-goal generation and arrival. The frontier selection policy is triggered upon reaching the last sub-goal, while the motion controller operates continuously at a fixed frequency.

III. Problem Statement and Method Overview

We focus on the problem of goal-reaching in unknown environments, in which the robot is tasked to reach a given position in previously unmapped regions. We look into randomly constructed environments composed of free regions R_f and occupied regions R_o. According to the task progress, the environment can be classified into known regions R_k and unknown regions R_u. The robot starts from a random position P_0 within the free region R_f and navigates towards the goal P_g located in initially unknown regions R_u. The known region R_k expands as the robot proceeds and the expanded regions are marked as free or occupied according to the sensor measurement. The objective of the task is to reach the goal P_g with the shortest possible path.

We present a goal-reaching system built on top of a graph-based map construction and hierarchical policy execution, as shown in Fig. 2. The path-extended graph is extracted from the roadmap and fed to the hierarchical policy as a compact map representation, recording the explored collision-free regions. The hierarchical pipeline is composed of a high-level frontier selection policy and a low-level motion controller handling path planning and collision avoidance. The frontier node with the most exploring value is selected by the DRL-based policy as a waypoint at each step and then passed to graph-based path-planning methods for sub-goal solving. As for navigation between sub-goals, a local navigation policy is adopted for collision avoidance according to the local gridmap. Detailed explanations are provided in Section V.

Note that the roadmap is updated incrementally as the navigation proceeds, while the gridmap is only maintained in the local scope. Traversability checking and node classification are also achieved through slope estimation and boundary detection within the local gridmap along the way.

IV. Map Representation: Path-Extended Graph

Memorizing explored regions' topological layouts and possible exploring directions are crucial for long-horizon goal-reaching in previously unknown environments. Thus, we introduce a path-extended graph as the map representation in a sufficient but concise style. We first build a roadmap graph $G_r^t = (V_r^t, E_r^t)$ with random sampling in the explored region. Lightweight loop closures are performed during the roadmap construction, enhancing the practicality for use in potentially large-scale applications. V_r^t and E_r^t refer to the roadmap node set and edge set respectively at time t. Each node in V_r^t is assigned with a feature vector $z_i^t = (x_i^t, y_i^t, c_i^t)$ when added to the graph, in which x_i^t and y_i^t indicate the position of the node in the roadmap coordinate system and c_i^t indicates the category of the node (*e.g.*, known, frontier, robot path, current robot position). The node features are continuously updated according to the position adjustment from loop closure and category changes during the progress. Traversability of the edges in E_r^t is checked between the nodes through slope estimation within the local gridmap. The roadmap graph G_r^t is built incrementally along the way and is updated effectively when map condition changes are observed. With this initial sparse topological map, we can further obtain the concise collision-free region representation for both task-relevant graph extraction and path planning.

To better enable the large-scale perception, we extract a path-extended graph $G_p^t = (V_p^t, E_p^t)$ from the initial roadmap G_r^t, keeping only necessary information. The path-extended graph node set V_p^t is composed of robot current node V_{robot}^t, path nodes V_{path}^t and frontier nodes $V_{frontier}^t$ derived from the current roadmap nodes V_r^t, meaning that $V_p^t = V_{robot}^t \cup V_{path}^t \cup V_{frontier}^t$. Instead of adding the goal as an independent node in the graph, we encode the relative goal information into node features. The node feature vectors in the path-extended graph are modified to ${z'}_i^t = (d_i^t, \theta_i^t, c_i^t)$, in which d_i^t refers to the relative distance between the goal position P_g and the node, θ_i^t refers to the relative angle between the goal position P_g and the node, and c_i^t still refers to the node category. Only the node category feature may change throughout an episode due to potential changes in the robot's position, while the goal information feature remains constant. The edges in E_p^t represent the traversability between the nodes in V_p^t, which not only can be obtained from the connections in G_r^t but also from the frontier updating procedure.

Path Node The original robot trajectory ψ is a sequence of history positions within the known region R_k, $\psi = (P_0, P_1, \ldots, P_t)$, $P_t \in R_k$. With the help of the roadmap, the trajectory points can be clustered to a subset of nodes

V_r^t according to the proximity of locations, providing both simplicity in recording structural information and prevention of redundant node expansion from multiple revisits to the same areas. The node subset V_{path}^t is continuously updated as the navigation proceeds, nodes newly marked as the robot position in V_r^t will be included and connected by checking the accessibility through the edge set E_r^t in the roadmap.

Frontier Node Frontiers are points on the boundaries between the known and unknown parts of the environment obtained from the local gridmap. Recent methods use sparse nodes with utility value [23] or nodes selected by human expert prediction [8] to describe the frontier distribution implicitly, which may impede the policy to understand the potential exploring directions along with the boundary layout. As we utilize a roadmap as the preliminary map representation, the initial frontiers can be further organized into nodes on the roadmap with the category of "frontier". We adopt frontier nodes $V_{frontier}^t$ derived from the roadmap node set V_r^t as frontiers, clearly representing the boundary information with low memory consumption. Each frontier node in $V_{frontier}^t$ stems from the path node that it is observed, describing the accessibility between the path node and the frontier node.

V. Policy Architecture: Graph-Based Reinforcement Learning

In this section, we aim to tackle long-horizon navigation in large-scale unknown environments with a hierarchical framework composed of a frontier selection module and a motion controller module. The frontier selection module determines the node with the most goal-reaching value as a high-level action. The motion controller module generates an optimal path covering the current robot node and the selected frontier node on the roadmap and then guides the robot to follow the given path while avoiding collisions.

A. Frontier Selection

The goal-reaching tasks in unknown environments can be decomposed into a series of frontier selection and arrival problems using partially explored maps. The optimal frontier is selected at each step by the policy to serve as the waypoint for the underlying motion controller, guiding the robot progressively toward the final goal. To achieve non-myopic decision-making with topology awareness, we employ deep reinforcement learning (DRL) to learn the policy with the path-extended graph. The goal-reaching in unknown environments can then be formulated as a Partially Observable Markov Decision Process in the reinforcement learning framework which can be described as a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \Omega, \mathcal{O}, \gamma)$, where $\mathcal{S}$ is the state space, representing the global environment information inaccessible to the robot in deployment, $\mathcal{A}$ is the action space, $\mathcal{P}$ is the transition probability, $\mathcal{R}$ is the reward function, Ω is the observation space and $\mathcal{O}$ is the observation probability conditioned on the actual state, γ is a discount factor describing the relative importance of future rewards.

Observation Space Our policy takes the current path-extended graph G_p^t at each time step t as input o_t, which contains sufficient information for frontier-level decision-making in goal-reaching tasks. The goal orientation and node category information are encoded in the node features, while the environmental structure information and frontier distribution details are encompassed within the topological relationships. All the node feature vectors are normalized to the range of $[-1, 1]$ before sending into the network.

Algorithm 1 Goal-reaching in Unknown Environments

Input: π_f, π_l: frontier selection policy and local policy;
Input: P_g, P_r^0: goal position and initial robot position;
Input: G_r^0: initial roadmap;
Input: N: re-plan length;
Initialize path-extended graph G_p^0 from the roadmap G_r^0;
repeat
  Update roadmap G_r^t according to sensor readings;
  Extract path-extended graph G_p^t from the roadmap G_r^t;
  Apply action $\pi_f(G_p^t)$ as the waypoint to follow;
  Plan path ψ^t to the waypoint on the roadmap G_r^t;
  Follow the path with π_l for N steps;
until $d < \delta_g$

Action Space All nodes with the category "frontier" in the path-extended graph G_p^t are considered potential actions at each time step t. We consider a stochastic selection mechanism in deployment. The policy network outputs the attention weights over those frontier nodes. The action frontier can be selected greedily or through probability converted from the attention weight distribution. The selected frontier will be sent to the motion controller as a waypoint for completion. To enhance the flexibility of the policy, in case the robot misses updated actions while approaching a distant frontier. we introduce a replan mechanism in the execution phase. The replan mechanism can also stabilize the training procedure for keeping step-wise rewards within a similar magnitude.

Reward Setup To guide the policy optimization, we have designed a reward function taking arriving target and shortening path distance into account:

$$R(s_t, a_t) = R_g + R_p$$

In particular, s_t and a_t refer to the state obtained from the simulation and action taken by the agent at time step t respectively. The policy gets R_g for arriving at the goal:

$$R_g = \begin{cases} r_{arrival} & \text{if goal reached} \\ 0 & \text{otherwise} \end{cases}$$

where we use goal-related rewards $r_{arrival}$ for guidance to avoid falling into local optimum that impedes the policy to find different completion routes in complex environments.

To ensure fast goal-reaching, it gets penalized with R_p :

$$R_p = w\mathcal{C}(s_t, a_t)$$

which defines a distance penalty weighted by coefficient w. $\mathcal{C}$ defines a cost function that considers the path length during the execution of a_t after s_t. Note that s_t is only accessible

during training. The policy relies solely on the observation o_t for decision-making in deployment.

Policy Network We adopt the deep reinforcement framework of discrete SAC for policy training, as shown in Fig. 2. The input path-extended graph G_p^t contains both task-related node features and connection correlations between the nodes. *Neighboring Aggregation* is performed initially for information exchange among all adjacent nodes. Graph Attention Networks (GATs) and Node Encoder composed of multiple Multi-Head-Attention layers are both used to handle this time-varying graph input for multi-scale perception. With these topology-aware node features, frontiers can get comprehensive perceptual information about the surrounding layout. Then the *Frontier Aggregation* phase further enhances features through a self-attention-based Frontier Encoder for global topological perception. Through querying from the current robot to all frontiers $V_{frontier}^t$ using a pointer layer [32], the actor network generates the attention weights over all potential action nodes in the *Action Evaluation* phase. The final action is chosen according to this weight distribution. The critic is a Graph Q Net which shares a similar structure with the actor network, except for the final layer that outputs an overall evaluation of the current path-extended graph.

B. Path Planning and Local Navigation

Frontiers selected work as waypoints that gradually guide the robot toward the goal position. The path planning module uses a graph-based A* algorithm to generate a path from the current node V_{robot}^t to the selected frontier location on the roadmap. Sub-goal is then derived from the path according to the replan mechanism threshold at each time step.

To safely follow the sub-goals, a motion policy is deployed for local dynamic collision avoidance using the methods in our previous works [33] [34]. The local policy takes laser readings in sequential obstacle map representation as input and outputs velocity commands. The policy is pre-trained separately in multi-agent simulations. The pseudo-code of the overall method is presented in Algorithm 1.

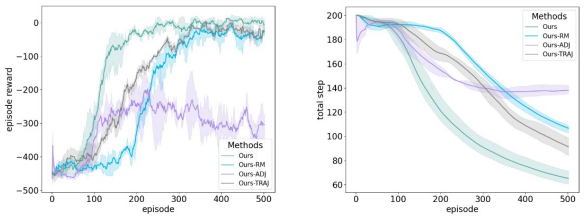
VI. Experiments

A. Experiment Setup

For efficient frontier policy training, we employ a dungeon map dataset [17]. The gridmaps are sampled at varying resolutions for different scales. Note that too low a resolution might hinder the representation of narrow areas, while too high a resolution could introduce unnecessary computational load. Roadmap points are randomly sampled within the robot's observable range and connected using Bresenham's line algorithm. A successful episode is defined by the robot being within $1m$ of the goal and below the maximum episode length. Environments are categorized into easy, medium, and complex levels as in [23]. Increased difficulty levels entail longer horizons of navigation and complex paths, raising the likelihood of encountering dead-ends.

Following metrics are used to compare the performance:

- ***Success Rate (SR):*** The ratio of arriving goals within the time allowed after 50 runs.



(a) Episode Reward **(b) Total Step**

Fig. 3. Ablation comparison on episode reward and total steps. We refer to our method as "Ours", the version using full roadmap input as "Ours-RM", the version using adjacent actions as "Ours-ADJ", and the version using trajectory graph as "Ours-TRAJ". The solid curves and shaded regions depict the mean and the standard deviation of the data among the three trials. We can observe that our method can achieve higher episode rewards and fewer total steps within fewer training episodes compared with other methods, indicating that our methods can efficiently handle environmental encoding and decision-making in complex environments.

- ***Success weighted by Path Length (SPL):*** The success rate weighted by normalized inverse path length.
- ***Average Map Size (Avg Map):*** Average map size during the navigation over the succeeded runs.

B. Ablation Study

Map Representation: To prove the merits of our map representation, we test DRL-based policy with inputs of different graph construction mechanisms in the dungeon map simulations, results are shown in Table I, Fig. 3 and Fig. 4.

- ***Ours-RM*** uses a full roadmap as input, containing nodes covering all the explored regions, which is also a randomly sampled version of the method used in [23].
- ***Ours-TRAJ*** uses a trajectory graph as input. Edges are constructed between temporally consecutive nodes or spatially neighboring nodes at each time step, recording the robot trajectories of the whole procedure [8].

We implement the modified versions by replacing the input but keeping the same range of the network receptive field for fair comparison. As the roadmaps or trajectory graphs with a large number of nodes may take considerable resource consumption with Transformer-based policy, we empirically set the range of the receptive field to 7. Results show that our method can achieve better performance in all scenarios with both higher rewards and less energy consumption, unnecessary movement is effectively avoided. We believe that the roadmap with massive intermediate connections brings redundant information, which can not be handled well by networks with a limited receptive field, especially in complex scenarios. The curves in Fig. 3 also show that the roadmap may lead to unstable performance and lower sample efficiency during training. The trajectory graph's edges are constructed considering local spatial correlations and global temporal navigation progress, which may cause ambiguities like graph-level distant nodes being spatially close. Our path-extended graph enables compact representation, facilitating efficient information aggregation within a reasonable receptive field. Performance enhancement becomes more pronounced with increasing difficulty levels, demonstrating our map representation's efficacy in large-scale perception.

Policy Architecture: We also evaluate policies with different action spaces and network structures:

(a) Utility(506) (b) CNN+LSTM (465) (c) Ours-RM (444) (d) Ours-ADJ (800) (e) Ours (224)

Fig. 4. Trajectory Comparison. A case example of the robot navigating from the purple point to the blue point in a previously unknown environment is shown here. The light gray parts represent explored regions while the dark gray parts represent unknown regions. Trajectory results show that our method can help the robot avoid back-and-forth behaviors within the explored regions and navigate through efficiently with large-scale topology perception. While other methods suffer from redundant movement or the trap of dead-ends.

TABLE I

MAP REPRESENTATION ABLATIONS.

Scenario	Methods \ Metrics	SR	SPL	Avg Map*
Easy	Ours-RM	1.0	0.84	585.3
	Ours-TRAJ	1.0	0.90	561.1
	Ours	**1.0**	**0.94**	**540.3**
Medium	Ours-RM	1.0	0.84	656.2
	Ours-TRAJ	1.0	0.85	650.1
	Ours	**1.0**	**0.89**	**642.2**
Complex	Ours-RM	0.96	0.73	682.3
	Ours-TRAJ	1.0	0.79	671.3
	Ours	**1.0**	**0.87**	**658.2**

* Calculated over the succeeded runs.

TABLE II

POLICY ARCHITECTURE ABLATIONS: ACTION SPACE

Scenario	Methods \ Metrics	SR	SPL	Avg Map*
Easy	Ours-ADJ	0.68	0.59	**539.1**
	Ours-Cluster	1.0	0.84	584.1
	Ours w/o Replan	1.0	0.82	588.5
	Ours	**1.0**	**0.94**	540.1
Medium	Ours-ADJ	0.54	0.44	646.6
	Ours-Cluster	1.0	0.83	652.7
	Ours w/o Replan	1.0	0.83	644.1
	Ours	**1.0**	**0.89**	**642.2**
Complex	Ours-ADJ	0.54	0.46	**647.2**
	Ours-Cluster	0.96	0.70	703.2
	Ours w/o Replan	1.0	0.79	668.2
	Ours	**1.0**	**0.87**	658.2

* Calculated over the succeeded runs.

- ***Ours-ADJ*** uses nodes adjacent to the current node as action space, within a distance matching our method's replan range to ensure comparable step distances.
- ***Ours-Cluster*** uses clustered frontiers as actions, with further sparsification by overlooking frontier layouts, while retaining the replanning mechanism.
- ***Ours w/o Replan*** uses the same policy network as ours but does not replan until the frontiers are reached.
- ***Ours-Isolated*** directly processes isolated frontiers with attention mechanisms, disregarding graph connections.
- ***Ours-PointNet*** uses a modified PointNet [35] to extract graph-level features, and concatenates them with node-level features for topology-awareness.

The results in Tables II and III demonstrate that our policy architecture outperforms others in efficient goal-reaching. The low success rate of ***Ours-ADJ*** highlights the effectiveness of using frontiers as actions, allowing the policy to focus on high-level decision-making instead of long-horizon planning. Improved sample efficiency compared to ***Ours-ADJ***, as shown in Fig. 3, further validates the effectiveness. The inferior ***Avg Map*** results suggest that ***Ours-ADJ*** struggles with complex tasks and is prone to dead-ends, as evidenced in Fig. 4. The reduced efficiency of the ***Ours-Cluster*** method emphasizes the significance of the frontier layout in action evaluation, where sparse directional guidance from clustered frontiers may cause ambiguity in the presence of multiple viable directions. The replanning mechanism's success, demonstrated by the results of ***Ours w/o Replan***, confirms its role in timely area recognition. Table III shows that the topological relationship is crucial for decision-making, necessitating a graph neural network for proper encoding and understanding. Overall, enhanced performance in complex scenarios validates the efficiency of our policy architecture for long-horizon goal-reaching tasks.

TABLE III

POLICY ARCHITECTURE ABLATIONS: NETWORK STRUCTURE.

Scenario	Methods \ Metrics	SR	SPL	Avg Map*
Easy	Ours-Isolated	1.0	0.82	591.2
	Ours-PointNet	1.0	0.88	567.3
	Ours	**1.0**	**0.94**	**540.3**
Medium	Ours-Isolated	1.0	0.81	675.2
	Ours-PointNet	1.0	0.82	665.5
	Ours	**1.0**	**0.89**	**642.2**
Complex	Ours-Isolated	0.96	0.69	708.1
	Ours-PointNet	0.94	0.65	733.1
	Ours	**1.0**	**0.87**	**658.2**

* Calculated over the succeeded runs.

C. Comparative Study

We conducted a performance evaluation of our method against various existing approaches. While some were developed for exploration and mapping, we reimplemented them for goal-reaching settings, results are shown in Table IV.

- ***Utility*** [18] uses an Information-based Distance Limited Exploration utility function for frontier evaluation, accounting for information gain and Euclidean distances.
- ***CNN+LSTM*** [4] uses a frontier selection policy that takes sequential occupancy maps and frontier locations as inputs and outputs a weight parameter that regulates the importance ratio of travel distance and goal distance.
- ***Graph Pooling*** uses graph-level pooling to distill features from the roadmap, and then maps them to the action space with eight uniformly sampled actions. Unreachable actions are masked via collision checks.

Methods utilizing frontiers as the actions demonstrate greater stability than ***Graph Pooling*** with adjacent actions. The pooling layer in ***Graph Pooling*** indiscriminately blends node information, hindering task-relevant feature extraction.

TABLE IV
COMPARISON.

Scenario	Methods \ Metrics	SR	SPL	Avg Map*
Easy	Utility	1.0	0.90	587.3
	CNN+LSTM	1.0	0.89	591.3
	Graph Pooling	0.64	0.54	**536.2**
	Ours	**1.0**	**0.94**	540.1
Medium	Utility	1.0	0.86	663.4
	CNN+LSTM	1.0	0.87	652.2
	Graph Pooling	0.51	0.41	648.3
	Ours	**1.0**	**0.89**	**642.2**
Complex	Utility	0.96	0.79	728.2
	CNN+LSTM	0.98	0.75	732.6
	Graph Pooling	0.45	0.36	**630.1**
	Ours	**1.0**	**0.87**	658.2

* Calculated over the succeeded runs.

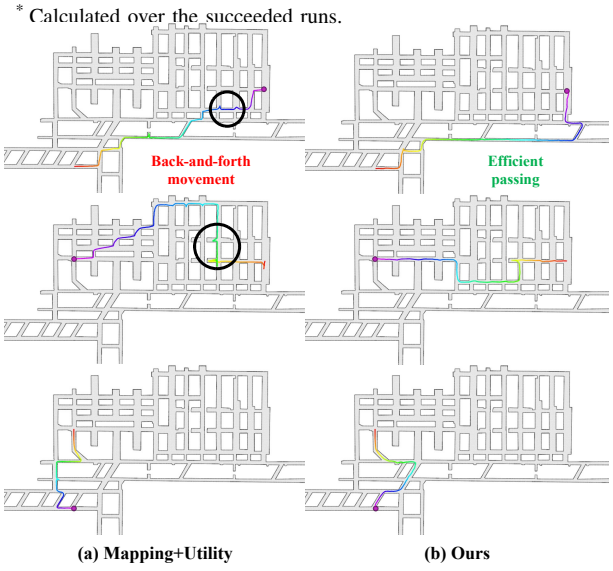


Fig. 5. Tunnel simulation comparison. The robot is tasked to sequentially navigate between three predefined targets with the ***Mapping+Utility*** method and our method. Trajectories show that our method can achieve a more efficient performance in complex long-horizon scenarios, avoiding back-and-forth behaviors. Quantitative results in Table V also prove so.

Frontiers offer consistent guidance, helping robots avoid local optima. However, ***Utility*** treats frontiers individually, neglecting their layout and interconnections, resulting in inefficient back-and-forth movements. ***CNN+LSTM*** slightly outperforms ***Utility*** but faces similar issues due to the recurrent neural network's catastrophic forgetting problem, limiting global perception. Our method, with its path-extended graph for concise large-scale perception and a policy focused on global decision-making, delivers the most effective and stable results. Its superior performance in complex settings highlights our method's proficiency in long-horizon tasks.

To demonstrate our method's effectiveness and efficiency, we compare it with a map-based approach in a tunnel simulation environment [36]:

- ***Mapping+Utility*** deploys dense mapping for environmental memory and utilizes a utility-based frontier policy considering both goal distance and path length.

In simulation tasks involving sequential navigation to three targets, TABLE V and Fig. 5 reveal that ***Mapping+Utility*** frequently explores unnecessary areas, resulting in higher explored volumes. Its extended time consumption stems from

TABLE V
TUNNEL SIMULATION COMPARISON.

Task	Methods \ Metrics	explored volume(m^3)	traveling distance(m)	time usage(s)
1∼2	Mapping+Utility	2553.4	**176.6**	335.5
	Ours	**2088.3**	202.5	**150.1**
2∼3	Mapping+Utility	3358.3	302.9	631,8
	Ours	**2471.8**	**171.2**	**108.1**
3∼1	Mapping+Utility	1382.4	90.9	133.9
	Ours	**1367.7**	**89.8**	**70.1**

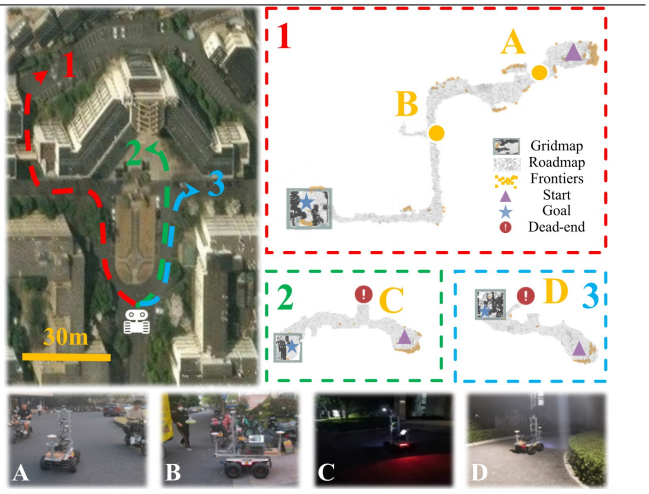


Fig. 6. Real world evaluation. We present three goal-reaching cases of our method in the previously unknown campus environment as shown in the satellite map marked with the colored path on the top left. The roadmap and the local gridmap along the way are shown on the top right. Sampled images on the right show some dynamic obstacle cases (A, B) or dead-end cases (C, D) during the process that are successfully handled by our method.

A* algorithm-based path-finding and repetitive movement due to its disregard for topology. In general, our method exhibits better performance in most long-horizon tunnel scenarios and effectively avoids redundant movement.

D. Real World Evaluation

Note that our method is platform and sensor agnostic, suitable for various mobile robots. We evaluated our method using LiDAR-equipped differential robots in an unmapped campus environment featuring non-uniform obstacles, uneven terrain, and dynamic entities like pedestrians and vehicles. The robots navigated to three targets with varying complexities: Task 1 involved long-horizon navigation with dead-end and dynamic obstacles during busy daytime hours; Tasks 2 and 3 entailed maneuvering through unstructured areas with stairs and slopes. As Fig. 6 indicates, the robots successfully reached goals while avoiding collisions. Efficient roadmap construction and motion control allowed the robots to concentrate on frontier-level decisions, navigating out of local dead-ends or incorrect paths. The success in these untrained campus environments demonstrates our method's adaptability and robustness. Videos can be found at https://www.bilibili.com/video/BV1rz421y7gp.

VII. CONCLUSION

In this study, we introduce a hierarchical method for goal-reaching, combining large-scale perception and long-horizon decision-making. Our path-extended graph facilitates effective perception in unknown environments, suitable for

learning-based methods with limited receptive field or computational resources. Our hierarchical architecture leverages deep reinforcement learning and a robust motion controller for topology-aware, long-horizon planning. This approach shows substantial improvement over existing methods in complex, unknown scenarios, validated through both simulation and real-world tests. Future work will focus on enhancing cooperative performance via joint training or dynamic-range feature encoding and exploring multi-robot tasks.

REFERENCES

[1] 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.

[2] B. 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'*. IEEE, 1997, pp. 146–151.

[3] H. Li, Q. Zhang, and D. Zhao, "Deep reinforcement learning-based automatic exploration for navigation in unknown environment," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 6, pp. 2064–2076, 2019.

[4] F. Niroui, K. Zhang, Z. Kashino, and G. Nejat, "Deep reinforcement learning robot for search and rescue applications: Exploration in unknown cluttered environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 610–617, 2019.

[5] E. Tolstaya, F. Gama, J. Paulos, G. Pappas, V. Kumar, and A. Ribeiro, "Learning decentralized controllers for robot swarms with graph neural networks," in *Conference on robot learning*. PMLR, 2020, pp. 671–682.

[6] F. Chen, J. Wang, T. Shan, and B. Englot, "Autonomous exploration under uncertainty via graph convolutional networks," in *The International Symposium of Robotics Research*. Springer, 2019, pp. 676–691.

[7] F. Chen, J. D. Martin, Y. Huang, J. Wang, and B. Englot, "Autonomous exploration under uncertainty via deep reinforcement learning on graphs," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 6140–6147.

[8] F. Chen, P. Szenher, Y. Huang, J. Wang, T. Shan, S. Bai, and B. Englot, "Zero-shot reinforcement learning on graphs for autonomous exploration under uncertainty," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 5193–5199.

[9] Z. Liu, Y. Zhai, J. Li, G. Wang, Y. Miao, and H. Wang, "Graph relational reinforcement learning for mobile robot navigation in large-scale crowded environments," *IEEE Transactions on Intelligent Transportation Systems*, 2023.

[10] J. Wang and B. Englot, "Autonomous exploration with expectation-maximization," in *Robotics Research: The 18th International Symposium ISRR*. Springer, 2020, pp. 759–774.

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

[12] Z. Ravichandran, L. Peng, N. Hughes, J. D. Griffith, and L. Carlone, "Hierarchical representations and explicit memory: Learning effective navigation policies on 3d scene graphs using graph neural networks," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 9272–9279.

[13] M. Lingelbach, C. Li, M. Hwang, A. Kurenkov, A. Lou, R. Martín-Martín, R. Zhang, L. Fei-Fei, and J. Wu, "Task-driven graph attention for hierarchical relational object navigation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 886–893.

[14] A. Francis, A. Faust, H.-T. L. Chiang, J. Hsu, J. C. Kew, M. Fiser, and T.-W. E. Lee, "Long-range indoor navigation with prm-rl," *IEEE Transactions on Robotics*, vol. 36, no. 4, pp. 1115–1134, 2020.

[15] T. Fan, X. Cheng, J. Pan, P. Long, W. Liu, R. Yang, and D. Manocha, "Getting robots unfrozen and unlost in dense pedestrian crowds," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1178–1185, 2019.

[16] J. A. Placed, J. Strader, H. Carrillo, N. Atanasov, V. Indelman, L. Carlone, and J. A. Castellanos, "A survey on active simultaneous localization and mapping: State of the art and new frontiers," *IEEE Transactions on Robotics*, 2023.

[17] F. Chen, S. Bai, T. Shan, and B. Englot, "Self-learning exploration and mapping for mobile robots via deep reinforcement learning," in *Aiaa scitech 2019 forum*, 2019, p. 0396.

[18] R. Cimurs, I. H. Suh, and J. H. Lee, "Goal-driven autonomous exploration through deep reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 730–737, 2021.

[19] T. Dang, F. Mascarich, S. Khattak, C. Papachristos, and K. Alexis, "Graph-based path planning for autonomous robotic exploration in subterranean environments," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 3105–3112.

[20] R. Reinhart, T. Dang, E. Hand, C. Papachristos, and K. Alexis, "Learning-based path planning for autonomous exploration of subterranean environments," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 1215–1221.

[21] D. Shah, B. Eysenbach, N. Rhinehart, and S. Levine, "Rapid exploration for open-world navigation with latent goal models," *arXiv preprint arXiv:2104.05859*, 2021.

[22] X. Xu, C. Wang, Y. Wang, and R. Xiong, "Hitmap: A hierarchical topological map representation for navigation in unknown environments," *arXiv preprint arXiv:2109.09293*, 2021.

[23] Y. Cao, T. Hou, Y. Wang, X. Yi, and G. Sartoretti, "Ariadne: A reinforcement learning approach using attention-based deep networks for exploration," *arXiv preprint arXiv:2301.11575*, 2023.

[24] W.-C. Lee, M. C. Lim, and H.-L. Choi, "Extendable navigation network based reinforcement learning for indoor robot exploration," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 11 508–11 514.

[25] Q. Li, F. Gama, A. Ribeiro, and A. Prorok, "Graph neural networks for decentralized multi-robot path planning," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 11 785–11 792.

[26] T. Gervet, S. Chintala, D. Batra, J. Malik, and D. S. Chaplot, "Navigating to objects in the real world," *Science Robotics*, vol. 8, no. 79, p. eadf6991, 2023.

[27] M. Keidar and G. A. Kaminka, "Efficient frontier detection for robot exploration," *The International Journal of Robotics Research*, vol. 33, no. 2, pp. 215–236, 2014.

[28] J. Zhang, L. Liu, P. Xiang, Q. Fang, X. Nie, H. Ma, J. Hu, R. Xiong, Y. Wang, and H. Lu, "Ai co-pilot bronchoscope robot," *Nature communications*, vol. 15, no. 1, p. 241, 2024.

[29] Q. Wu, J. Wang, J. Liang, X. Gong, and D. Manocha, "Image-goal navigation in complex environments via modular learning," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 6902–6909, 2022.

[30] Y. F. Chen, M. Everett, M. Liu, and J. P. How, "Socially aware motion planning with deep reinforcement learning," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 1343–1350.

[31] D. S. Chaplot, R. Salakhutdinov, A. Gupta, and S. Gupta, "Neural topological slam for visual navigation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 12 875–12 884.

[32] O. Vinyals, M. Fortunato, and N. Jaitly, "Pointer networks," *Advances in neural information processing systems*, vol. 28, 2015.

[33] Y. Cui, H. Zhang, Y. Wang, and R. Xiong, "Learning world transition model for socially aware robot navigation," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 9262–9268.

[34] Y. Cui, L. Lin, X. Huang, D. Zhang, Y. Wang, W. Jing, J. Chen, R. Xiong, and Y. Wang, "Learning observation-based certifiable safe policy for decentralized multi-robot navigation," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 5518–5524.

[35] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660.

[36] C. Cao, H. Zhu, F. Yang, Y. Xia, H. Choset, J. Oh, and J. Zhang, "Autonomous exploration development environment and the planning algorithms," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 8921–8928.