

Graph-based SLAM-Aware Exploration with Prior Topo-Metric Information

Ruofei Bai^{1,2}, Hongliang Guo², Member, IEEE, Wei-Yun Yau², Senior Member, IEEE, Lihua Xie¹, Fellow, IEEE

Abstract—Autonomous exploration requires a robot to explore an unknown environment while constructing an accurate map using Simultaneous Localization and Mapping (SLAM) techniques. Without prior information, the exploration performance is usually conservative due to the limited planning horizon. This paper exploits prior information about the environment, represented as a topo-metric graph, to benefit both the exploration efficiency and the pose graph reliability in SLAM. Based on the relationship between pose graph reliability and graph topology, we formulate a SLAM-aware path planning problem over the prior graph, which finds a fast exploration path enhanced with the globally informative loop-closing actions to stabilize the SLAM pose graph. A greedy algorithm is proposed to solve the problem, where theoretical thresholds are derived to significantly prune non-optimal loop-closing actions, without affecting the potential informative ones. Furthermore, we incorporate the proposed planner into a hierarchical exploration framework, with flexible features including path replanning, and online prior graph update that adds additional information to the prior graph. Simulation and real-world experiments indicate that the proposed method can reliably achieve higher mapping accuracy than compared methods when exploring environments with rich topologies, while maintaining comparable exploration efficiency. Our method has been open-sourced on GitHub.

Index Terms—Planning under uncertainty, SLAM, autonomous exploration

I. INTRODUCTION

Autonomous exploration has been deemed as one of the important tasks for robotics, with applications to surveillance and inspection, search and rescue, etc. Typically, a robot uses onboard sensors such as LiDAR or cameras to perceive an unknown environment and constructs an accurate map for downstream tasks, known as Simultaneous Localization and Mapping (SLAM). In pose graph-based SLAM methods, the map can be constructed by registering onboard sensor readings at each pose into a global coordinate frame [1], [2]. To get an accurate map of the environment, the robot should maintain accurate and reliable pose estimation during the exploration.

However, the pose estimation may accumulate drift as the robot explores the environment due to noise or environment degeneracy. Active SLAM methods [3] are used to drive the robot to actively revisit some places to establish relative measurements, called *loop closures*, to mitigate the cumulative drift in SLAM. Exploration actions are usually selected based

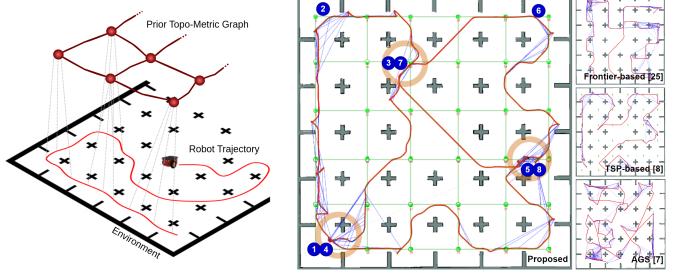


Fig. 1. SLAM-Aware exploration with prior topo-metric graph. The right figures show the occupancy grid map built after exploration, the prior topo-metric graph (green), the SLAM pose graph with odometry edges (red) and loop closures (blue). The numbers in blue circles indicate the order in which the robot visits different regions. Compared with other methods, the exploratory trajectory of the proposed method forms three big loops that quickly cover the environment while forming a globally reliable pose graph.

on their information gain, considering both the environment coverage and the robot pose uncertainty [4]–[7]. However, due to the lack of prior information about the environment, they balance the two objectives only within a local planning horizon, thus being conservative in both exploration efficiency and finding informative loop closures. In fact, the locally selected loop closures contribute less to mitigating the drift since the odometry drift increases with distance.

In this paper, we use the prior topological and metric information of the environment to improve the exploration performance of a robot. Such information can be obtained from hand-drawn maps, outdated floorplans, or previously built maps [8], [9], and represented as a topo-metric graph to describe the positions of regions and their connectivity in the environment. Based on the relationship between the pose graph reliability and the graph topology [10]–[12], we formulate a SLAM-aware path planning problem over the prior graph, which finds a high-level path to guide the robot to quickly cover the environment while forming a well-connected pose graph for reliable SLAM estimation. With the prior topo-metric graph, both exploration efficiency and pose graph reliability are effectively balanced from a global perspective compared to reactive methods, as illustrated in Fig. 1.

The contributions of this paper are summarized as follows: 1) A SLAM-aware path planner over a prior topo-metric graph that explicitly considers both exploration efficiency and pose graph reliability. We derive theoretical thresholds that significantly prune non-optimal loop-closing actions to speed up an iterative greedy algorithm; 2) A flexible hierarchical exploration framework that integrates the SLAM-aware path planner, with several features including path replanning, and

¹Ruofei Bai and Lihua Xie are with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798 (e-mail: ruofei001@ntu.edu.sg; elhxie@ntu.edu.sg)

²Ruofei Bai, Hongliang Guo and Wei-Yun Yau are with the Institute for Infocomm Research (I2R), Agency for Science, Technology and Research (A*STAR), Singapore 138632 (e-mail: stubair@i2r.a-star.edu.sg; guo_hongliang@i2r.a-star.edu.sg; wyyau@i2r.a-star.edu.sg)

online prior graph update that adds additional information to the prior graph. Moreover, simulation and real-world experiments verify the effectiveness of the proposed method in various environments, where the proposed method reliably maintains higher mapping accuracy and comparable exploration efficiency than the compared methods.

II. RELATED WORKS

This section reviews the related works in autonomous exploration exploiting the prior information and recent works in relating pose graph uncertainty with the graph topology.

A. Autonomous Exploration based on Prior Information

In autonomous exploration, the robot usually applies active SLAM methods to decide its exploratory actions to balance the SLAM accuracy and the exploration efficiency. A detailed review of active SLAM can be found in [3].

In some applications, prior information about the environment is available, such as the floor plan, a topological sketch, etc. Several works utilizing prior information have been proposed to improve exploration performance. Luperto *et al.* [13] use a floor plan of the environment to help more accurately evaluate the information gain of each frontier in the exploration. Oßwald *et al.* [8] use a prior topo-metric graph of the environment to speed up the exploration, where a traveling salesman problem (TSP) solver finds a path over the prior topo-metric graph to guide the exploration behavior of the robot. However, the pose uncertainty and map accuracy are not considered. A similar situation is considered in [14], where a robot is required to traverse all edges of a prior topo-metric graph of the environment. The distance of a robot's future path is scaled by the robot's pose uncertainty, which motivates the robot to actively close loops to reduce its pose uncertainty. Xue *et al.* [9] further consider the active SLAM over a prior topo-metric graph with the starting position unknown. However, they do not consider the pose uncertainty during the exploration. Different from existing works, this paper considers both the exploration efficiency and the pose graph reliability in a SLAM-aware path planning formulation, making full use of the prior topo-metric information of the environment.

B. Graph-based SLAM Uncertainty Evaluation

In graph-based SLAM methods [1], it is important to maintain a reliable pose graph, which reflects the accuracy of the built map in the SLAM process [15]. To efficiently quantify the reliability of the SLAM pose graph, an increasing number of works including [10], [11], [16] and [7], [12] use the graph topology metric to evaluate the estimation uncertainty in the pose graph optimization. Specifically, Khosoussi *et al.* [10] rearrange the FIM into the translational part and the rotational part and prove that each part is individually related to the pose graph Laplacian matrix weighted by the corresponding covariance matrices. Chen *et al.* [11] further extend this work into 3D case. On the other hand, Placed *et al.* [12] directly encapsulate the covariance matrix of each observation into

the edge weight of the Laplacian matrix to more efficiently evaluate the pose graph reliability.

The above topology-based metrics have been applied to evaluate potential loop closures in robot exploration [7], [17]. However, the metric is either evaluated in a reactive manner, or limited in frontier evaluation. Instead, we use the above relationship to find a distance-efficient path with globally informative loop closures over the prior topo-metric graph, which guides the robot to form a well-connected pose graph for reliable SLAM estimation.

III. PRELIMINARIES

A. Graph Laplacian

Given a connected graph \mathcal{G} with $n+1$ vertices and m edges, the graph Laplacian matrix is defined as:

$$\mathbf{L}^\circ = \mathbf{B}^\circ \mathbf{B}^{\circ\top} = \sum_{j=1}^m \mathbf{B}_j \mathbf{B}_j^\top \in \mathbb{R}^{(n+1) \times (n+1)}, \quad (1)$$

where $\mathbf{B}^\circ \in \mathbb{R}^{(n+1) \times m}$ is the incidence matrix of \mathcal{G} , \mathbf{B}_j is the j -th column vector of \mathbf{B}° that only has non-zero values at two indices corresponding to the vertices connected by the j -th edge. Each row of \mathbf{B}° corresponds to a vertex and each column corresponds to an edge in \mathcal{G} . The weighted Laplacian matrix of the graph \mathcal{G} is defined as $\mathbf{L}_\gamma = \sum_{j=1}^m \gamma_j \mathbf{B}_j \mathbf{B}_j^\top$, where γ_j is the weight of j -th edge in \mathcal{G} .

If one vertex is anchored in \mathcal{G} , i.e., the corresponding row is removed from the incidence matrix \mathbf{B}° , the obtained Laplacian matrix is called the reduced Laplacian matrix, denoted as $\mathbf{L} \in \mathbb{R}^{n \times n}$. The weighted reduced Laplacian matrix \mathbf{L}_γ can also be defined similarly.

B. Relating Pose Graph Reliability with Graph Laplacian

In graph-based SLAM algorithms [1], a pose graph is incrementally built, denoted as $\mathcal{G}^{\text{pose}} = \langle \mathcal{X}, \mathcal{Z} \rangle$, where each node in \mathcal{X} corresponds to a robot pose, and an edge $z_{ij} \in \mathcal{Z}$ represents the relative motion between two poses x_i and x_j , established by either odometry measurements or loop closure detection. The pose graph optimization aims to find the maximum likelihood estimation (MLE) of all poses \mathcal{X} , given all relative observations \mathcal{Z} . The objective function in pose graph optimization is defined as $\min_{\mathcal{X}} \mathbf{F}(\mathcal{X}) = \sum_{z_{ij} \in \mathcal{Z}} \mathbf{F}_{ij}(\mathcal{X}) = \sum_{z_{ij} \in \mathcal{Z}} \mathbf{e}_{ij}^\top \Sigma_{ij}^{-1} \mathbf{e}_{ij}$, where $\mathbf{e}_{ij} = z_{ij} - \hat{z}_{ij}(x_i, x_j)$ is the residual error between the predicted relative observation \hat{z}_{ij} and the actual observation z_{ij} ; Σ_{ij} is the covariance matrix of z_{ij} . The poses in \mathcal{X} can be optimized using nonlinear optimization methods like the Gauss-Newton method. The Hessian matrix \mathbf{H} in pose graph optimization is derived as

$$\mathbf{H} = \frac{1}{2} \sum_{z_{ij} \in \mathcal{Z}} \mathbf{J}_{ij}^\top \Sigma_{ij}^{-1} \mathbf{J}_{ij} = \frac{1}{2} \sum_{z_{ij} \in \mathcal{Z}} \mathbf{B}_{ij} \mathbf{B}_{ij}^\top \otimes \tilde{\Sigma}_{ij}^{-1}, \quad (2)$$

where \mathbf{J}_{ij} is the Jacobian matrix of the error function \mathbf{F}_{ij} w.r.t. \mathcal{X} , $\tilde{\Sigma}_{ij}$ is the covariance matrix of the observation z_{ij} with coordinate transformation, and $\mathbf{B}_{ij} \mathbf{B}_{ij}^\top$ is the Laplacian factor of the edge z_{ij} in the pose graph. As \mathbf{F}_{ij} only relates to x_i and x_j , the corresponding \mathbf{J}_{ij} only has non-zero values at indices i and j , which shares a similar structure as the column vector \mathbf{B}_{ij} of the incidence matrix of the pose graph. The details of the derivation are referred to [12].

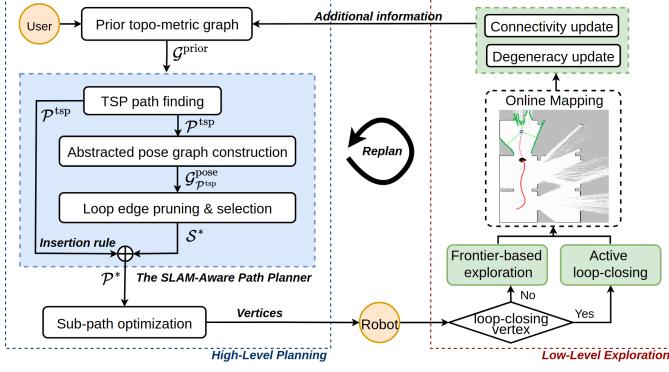


Fig. 2. The diagram of the proposed hierarchical exploration framework, including high-level planning and low-level navigation. The SLAM-aware path planner takes the prior topo-metric graph $\mathcal{G}^{\text{prior}}$ from users and outputs a high-level path \mathcal{P}^* . The robot follows vertices in \mathcal{P}^* to perform either active loop-closing or frontier-based exploration within a region. The prior graph is updated online, based on which the replanning and sub-path optimization modules opportunistically find a better path for exploration.

The Hessian matrix is also known as the *observed FIM* [16], denoted as $\mathbb{I}(\mathcal{X})$. In practice, the observed FIM $\mathbb{I}(\mathcal{X})$ computed at the MLE of \mathcal{X} is quantified to reflect the uncertainty of \mathcal{X} , using scalar functions of $\mathbb{I}(\mathcal{X})$, among which the D-optimal is shown to be superior in capturing the global uncertainty in all dimensions of the poses [18]. The D-optimal is a scalar function from Theory of Optimal Experimental Design [19] and is defined as $D\text{-opt}(\mathbf{L}) = \det(\mathbf{L})^{\frac{1}{n}}$ for an $n \times n$ matrix. According to [12], the D-optimal of $\mathbb{I}(\mathcal{X})$ can be approximated by:

$$\begin{aligned} D\text{-opt}(\mathbb{I}(\mathcal{X})) &= D\text{-opt}\left(\sum_{z_{ij} \in \mathcal{Z}} \mathbf{B}_{ij} \mathbf{B}_{ij}^\top \otimes \tilde{\Sigma}_{ij}^{-1}\right) \\ &\approx D\text{-opt}\left(\sum_{z_{ij} \in \mathcal{Z}} \mathbf{B}_{ij} \mathbf{B}_{ij}^\top \cdot D\text{-opt}(\tilde{\Sigma}_{ij}^{-1})\right) = D\text{-opt}(\mathbf{L}_\gamma). \end{aligned} \quad (3)$$

The subscript γ in \mathbf{L}_γ indicates that the reduced Laplacian matrix of the pose graph is weighted by the D-optimal of the inverse of the covariance matrices for each edge in the pose graph. Compared with the Hessian matrix \mathbf{H} , the weighted Laplacian matrix \mathbf{L}_γ has smaller dimensions that only depend on the number of poses, and is more efficient in evaluating the pose graph reliability. Note that Eq. (3) establishes the relationship between the pose graph reliability and graph topology, and the latter will be used in this paper for SLAM uncertainty evaluation.

IV. SLAM-AWARE PATH PLANNING OVER PRIOR GRAPH

The key idea of this paper is to find a high-level path over the prior topo-metric graph of the environment, to guide the low-level exploration behavior of the robot to achieve quick coverage while forming a well-connected pose graph for reliable SLAM estimation. The proposed hierarchical exploration framework is shown in Fig. 2 and will be introduced in detail in Sec. V. This section focuses on the high-level planning part. We formulate the SLAM-aware path planning problem based on the prior graph, and propose the greedy algorithm with effective pruning mechanisms to solve it.

A. Representation of the Prior Topo-Metric Information

The prior knowledge about the coarse structure of an environment can be represented by a prior topological-metric graph, defined as $\mathcal{G}^{\text{prior}} = \langle \mathcal{V}, \mathcal{E}, \omega \rangle$, where $\mathcal{V} \subseteq \mathbb{R}^2 / \mathbb{R}^3$ includes set of vertices distributed in the environment, where each vertex corresponds to a convex region to be explored, like room or corridor, etc. Generally, one vertex per convex region is sufficient [8]. The set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ includes all edges, where an edge $(v_i, v_j) \in \mathcal{E}$ indicates there exists a traversable path between v_i and v_j . $\omega(v_i, v_j)$ denotes the distance between v_i and v_j in $\mathcal{G}^{\text{prior}}$. We assume the prior graph is connected, and the starting pose of the robot w.r.t. $\mathcal{G}^{\text{prior}}$ is given initially. The starting vertex of the robot is denoted as v_{start} .

B. Two-Stage Strategy for SLAM-Aware Path Planning

Given the prior topo-metric graph $\mathcal{G}^{\text{prior}}$, the problem is to find a walk \mathcal{P} over $\mathcal{G}^{\text{prior}}$ that covers all vertices in $\mathcal{G}^{\text{prior}}$, while forming a well-connected pose graph topology for reliable SLAM estimation. The problem is similar to solving a traveling salesman problem (TSP), which finds the shortest path to visit each vertex in a graph once and return to the starting vertex, and it is NP-hard to solve [20]. While the TSP only considers the distance cost, we need to further consider the reliability of the pose graph induced by the path. To reduce the computational complexity, we propose the following two-stage strategy to solve the problem:

- (1) Find an exploration path starting from v_{start} to quickly cover all vertices in $\mathcal{G}^{\text{prior}}$, where existing TSP solvers can be used;
- (2) Select and insert informative and distance-efficient loop-closing actions along the exploration path to improve the pose graph reliability.

Specifically, to construct a TSP instance in Stage (1), the prior graph $\mathcal{G}^{\text{prior}}$ is first transformed into a complete graph, where the inter-vertex distance is defined as the shortest path length between the two vertices in $\mathcal{G}^{\text{prior}}$. Existing TSP solvers can then be used to find the shortest path over the complete graph starting from v_{start} to visit all vertices in $\mathcal{G}^{\text{prior}}$ exactly once. Note the original TSP definition has additional returning constraints, which can be eliminated by resetting the distance of all incoming edges of v_{start} as zero and deleting the final vertex in the return TSP path, which is also known as Open-loop TSP [21]. After obtaining the TSP path, we connect consecutive vertices in the path with the shortest path between them in $\mathcal{G}^{\text{prior}}$, to recover the actual path that follows the connectivity constraints of $\mathcal{G}^{\text{prior}}$. The finally obtained path, which is actually a walk over $\mathcal{G}^{\text{prior}}$, is denoted as $\mathcal{P}^{\text{tsp}} = (v_{i_1}, \dots, v_{i_{|\mathcal{P}^{\text{tsp}}|}})$, where $\forall k \in [1 : |\mathcal{P}^{\text{tsp}}| - 1]$, $v_{i_k} \in \mathcal{V}$, $(v_{i_k}, v_{i_{k+1}}) \in \mathcal{E}$; and $v_{i_1} = v_{\text{start}}$. The walk \mathcal{P}^{tsp} visits all vertices in \mathcal{V} and ensures complete coverage of the environment.

C. Loop Edge Selection over Abstracted Pose Graph

As the obtained walk \mathcal{P}^{tsp} only aims to quickly cover vertices in $\mathcal{G}^{\text{prior}}$, Stage (2) further improves the resulting pose graph reliability of the path, by adding informative and

distance-efficient loop-closing actions along \mathcal{P}^{tsp} . However, while the exploration efficiency can be directly quantified by the distance cost of \mathcal{P}^{tsp} , the pose graph reliability can not be evaluated directly without the actual pose graph constructed in SLAM. On the positive side, in our proposed framework, the walk \mathcal{P}^{tsp} over $\mathcal{G}^{\text{prior}}$ captures the abstracted structure of the robot's actual pose graph, because the robot will follow the high-level guidance of \mathcal{P}^{tsp} to explore the environment. Therefore, we define the *abstracted* pose graph from \mathcal{P}^{tsp} to provide a high-level approximation of the robot's actual pose graph, which will be used to identify informative loop-closing actions along the walk \mathcal{P}^{tsp} to reducing the SLAM uncertainty.¹

Definition 1 (Abstracted pose graph): Given a walk \mathcal{P} over a graph $\mathcal{G}^{\text{prior}} = \langle \mathcal{V}, \mathcal{E}, \omega \rangle$, an abstracted pose graph corresponding to \mathcal{P} is defined as $\mathcal{G}_{\mathcal{P}}^{\text{pose}} = \langle \mathcal{X}, \mathcal{Z} \rangle$, where $\mathcal{X} \subseteq \text{SE}(2)/\text{SE}(3)$ and $\mathcal{Z} \subseteq \mathcal{X} \times \mathcal{X}$. The poses in \mathcal{X} have a one-to-one correspondence with vertices in \mathcal{V} covered by \mathcal{P} , and an edge $(x_i, x_j) \in \mathcal{Z}$ exists if the edge $(\mathcal{M}(x_i), \mathcal{M}(x_j)) \in \mathcal{E}$ and is covered by \mathcal{P} , where $\mathcal{M} : \mathcal{X} \rightarrow \mathcal{V}$ is a mapping function from \mathcal{X} to \mathcal{V} . The subscripts of the poses in \mathcal{X} are assigned according to the order in which their corresponding vertices are visited by \mathcal{P} .

By following Def. 1, the abstracted pose graph corresponding to \mathcal{P}^{tsp} is constructed and denoted as $\mathcal{G}_{\mathcal{P}^{\text{tsp}}}^{\text{pose}}$. The repeated visits of vertices and edges in $\mathcal{G}^{\text{prior}}$ by \mathcal{P}^{tsp} are only counted once for simplicity, so $\mathcal{G}_{\mathcal{P}^{\text{tsp}}}^{\text{pose}}$ has the same topology as a subgraph of $\mathcal{G}^{\text{prior}}$ covered by \mathcal{P}^{tsp} . Note that the orientation of poses is not explicitly defined in Def. 1, because the exact values of pose orientations do not affect the evaluation of SLAM uncertainty, as suggested in Eq. (3).

Remark 1: If the edge lengths in the original $\mathcal{G}^{\text{prior}}$ vary significantly, we can add additional vertices along the long edges in $\mathcal{G}^{\text{prior}}$ to make them balanced for better approximation of the abstracted pose graph. Meanwhile, the connectivity information in $\mathcal{G}^{\text{prior}}$ remains unchanged.

With the abstracted pose graph $\mathcal{G}_{\mathcal{P}^{\text{tsp}}}^{\text{pose}}$, we then identify candidate loop-closing actions, called *loop edges*, along the walk \mathcal{P}^{tsp} , and select the most informative and distance-efficient ones to be inserted into the walk to improve the pose graph reliability. The candidate loop edges in $\mathcal{G}_{\mathcal{P}^{\text{tsp}}}^{\text{pose}}$ are defined as follows:

Definition 2 (Loop edge): A loop edge connects two poses that are not directly connected in $\mathcal{G}_{\mathcal{P}^{\text{tsp}}}^{\text{pose}}$. Given $\mathcal{G}_{\mathcal{P}^{\text{tsp}}}^{\text{pose}} = \langle \mathcal{X}, \mathcal{Z} \rangle$, the set of all candidate loop edges is defined as $\mathcal{S} = \{z_{ij} = (x_i, x_j) \mid x_i, x_j \in \mathcal{X}; i > j; z_{ij} \notin \mathcal{Z}\}$.

A loop edge is not a loop closure, but a continuous movement to connect two poses in the abstracted pose graph, to improve the pose graph reliability. If a loop edge (x_i, x_j) is selected, the vertex $\mathcal{M}(x_i)$ in \mathcal{P}^{tsp} will be replaced by $(\mathcal{M}(x_i), \mathcal{M}(x_j), \mathcal{M}(x_i))$, and we refer this operation as *insertion rule*. It means that after the robot covers $\mathcal{M}(x_i)$, it will move from $\mathcal{M}(x_i)$ to $\mathcal{M}(x_j)$ to establish loop closures and then return to $\mathcal{M}(x_i)$ to continue exploration. The vertex

¹A similar idea of high-level abstraction of the pose graph is proposed in hierarchical pose graph optimization [22], where the poses in the robot's actual pose graph are clustered to construct an abstracted pose graph, and then only the abstracted pose graph is optimized to facilitate efficiency.

$\mathcal{M}(x_j)$ is called a loop-closing vertex. With such, the extra distance cost introduced by the loop edge (x_i, x_j) can be conveniently counted as $2 \cdot \omega(z_{ij})$, here we abuse the notation $\omega(\cdot)$ and let $\omega(z_{ij})$ denotes the shortest distance between $\mathcal{M}(x_i)$ and $\mathcal{M}(x_j)$ in $\mathcal{G}^{\text{prior}}$. Note that in actual exploration, the robot will skip previously visited vertices as introduced in Sec. V-A.

The loop edge selection problem is formulated as follows:

Problem 1: Given a prior topo-metric graph $\mathcal{G}^{\text{prior}}$, an initial walk $\mathcal{P}^{\text{tsp}} = (v_{i_1}, \dots, v_{i_{|\mathcal{P}^{\text{tsp}}|}})$ over $\mathcal{G}^{\text{prior}}$, and a set \mathcal{S} of candidate loop edges, find a subset $\mathcal{S}' \subseteq \mathcal{S}$ that maximize the following objective function:

$$\mathcal{J}(\mathcal{S}') = \frac{D\text{-opt} \left(\mathbf{L}_\gamma(\mathcal{P}^{\text{tsp}}) + \sum_{z_{ij} \in \mathcal{S}'} \gamma_{ij} \cdot \mathbf{B}_{ij} \mathbf{B}_{ij}^\top \right)}{D(\mathcal{P}^{\text{tsp}}) + 2 \cdot \sum_{z_{ij} \in \mathcal{S}'} \omega(z_{ij})},$$

where $\mathbf{L}_\gamma(\mathcal{P}^{\text{tsp}})$ is the reduced Laplacian matrix of the abstracted pose graph $\mathcal{G}_{\mathcal{P}^{\text{tsp}}}^{\text{pose}}$; $\mathbf{B}_{ij} \mathbf{B}_{ij}^\top$ is the Laplacian factor corresponding to the loop edge z_{ij} in $\mathcal{G}_{\mathcal{P}^{\text{tsp}}}^{\text{pose}}$; γ_{ij} is the weight of the edge z_{ij} ²; $D(\mathcal{P}^{\text{tsp}}) = \sum_{k=1}^{|\mathcal{P}^{\text{tsp}}|-1} \omega(v_{i_k}, v_{i_{k+1}})$ is the distance cost of \mathcal{P}^{tsp} ; and $2 \cdot \sum_{z_{ij} \in \mathcal{S}'} \omega(z_{ij})$ is the extra distance cost introduced by the selected loop edges in \mathcal{S}' .

The Problem 1 aims to find loop edges that balance the exploration distance and the pose graph reliability, with the numerator of the objective function $\mathcal{J}(\cdot)$ being the topology metric that quantifies the estimation uncertainty in SLAM, when adding several loop edges into the abstracted pose graph. An intuitive interpretation of $\mathcal{J}(\cdot)$ is that the extra distance traveled by the robot must contribute to reducing the averaged uncertainty of the pose graph, otherwise the loop-closing actions will not be performed. Here the TSP path \mathcal{P}^{tsp} provides a natural lower bound for the objective function $\mathcal{J}(\cdot)$ for further improvement. The algorithm to solve Problem 1 will be introduced in the next subsection.

D. Greedy Solution with Pruning Thresholds

This section presents a greedy-based iterative algorithm to solve Problem 1. We derive the pruning thresholds to speed up the greedy algorithm, which prune non-optimal candidate loop edges beforehand without affecting the informative ones. The proposed greedy algorithm with the pruning mechanism is summarized in Alg. 1.

Specifically, in each iteration, the greedy algorithm selects the loop edge that has the maximum contribution to the objective function \mathcal{J} , adds it into \mathcal{S}' , and then re-computes the contribution of all remaining candidate edges based on the updated \mathcal{S}' . The algorithm terminates until no candidate loop edge can further improve the objective value. The greedy algorithm is time-consuming as it needs to re-evaluate the contribution of all candidate edges in each iteration. Assume we already have a set \mathcal{S}' of selected loop edges, and let \mathcal{P}' denote the obtained path by inserting the selected loop edges

²In initial prior graph, all edges have the same covariance matrices, and will be updated online during exploration, as introduced in Sec. V-A. The initial value is set as $\text{Diag}\{0.1m, 0.1m, 0.001\text{rad}\}$ in the 2D experiments.

into \mathcal{P}^{tsp} following the insertion rule. Now consider adding another loop edge z_{ij} into \mathcal{S}' , the objective will be:

$$\begin{aligned}\mathcal{J}(\mathcal{S}' \cup \{z_{ij}\}) &= \frac{D\text{-opt}(\mathbf{L}_\gamma(\mathcal{P}') + \gamma_{ij} \cdot \mathbf{B}_{ij} \mathbf{B}_{ij}^\top)}{D(\mathcal{P}') + 2\omega(z_{ij})} \\ &= \mathcal{J}(\mathcal{S}') \cdot \frac{(1 + \gamma_{ij} \cdot \mathbf{B}_{ij}^\top \mathbf{L}_\gamma^{-1}(\mathcal{P}') \mathbf{B}_{ij})^{\frac{1}{n}}}{(1 + \frac{2\omega(z_{ij})}{D(\mathcal{P}')})} = \mathcal{J}(\mathcal{S}') \cdot \delta(z_{ij}, \mathcal{P}').\end{aligned}$$

Here the incremental term $\delta(z_{ij}, \mathcal{P}')$ quantifies the contribution of the loop edge z_{ij} to the objective \mathcal{J} . It depends on the current \mathcal{P}' that changes in each iteration. As introduced before, the set \mathcal{S} includes edges with the order of $\mathcal{O}(|\mathcal{V}|^2)$. If the greedy algorithm terminates in k iterations, we have to evaluate edges of order $\mathcal{O}(k \cdot |\mathcal{V}|^2)$ in total. To reduce the time complexity, we establish the following proposition to prune non-optimal loop edges from \mathcal{S} .

Proposition 1: Under the assumption $D(\mathcal{P}') \leq 2 \cdot D(\mathcal{P}^{\text{tsp}})$ ³, if a loop edge $z_{ij} \in \mathcal{S}$ satisfies

$$\frac{(1 + \gamma_{ij} \cdot \mathbf{B}_{ij}^\top \mathbf{L}_\gamma^{-1}(\mathcal{P}^{\text{tsp}}) \mathbf{B}_{ij})^{\frac{1}{n}}}{1 + \frac{\omega(z_{ij})}{D(\mathcal{P}^{\text{tsp}})}} \leq 1, \quad (4)$$

then for $\mathcal{S}' \subseteq \mathcal{S}$, it holds that $\mathcal{J}(\mathcal{S}' \cup \{z_{ij}\}) \leq \mathcal{J}(\mathcal{S}')$.

Proof 1: Since the reduced Laplacian matrix \mathbf{L}_γ is positive definite, its inverse \mathbf{L}_γ^{-1} is also positive definite. According to Lemma 9 of [10], for two positive definite matrix $\mathbf{L}_\gamma(\mathcal{P}')$ and $\mathbf{L}_\gamma(\mathcal{P}^{\text{tsp}})$, $\mathbf{L}_\gamma(\mathcal{P}') \succeq \mathbf{L}_\gamma(\mathcal{P}^{\text{tsp}})$ iff $\mathbf{L}_\gamma^{-1}(\mathcal{P}^{\text{tsp}}) \succeq \mathbf{L}_\gamma^{-1}(\mathcal{P}')$. Since \mathcal{P}^{tsp} is a subgraph of \mathcal{P}' , we have $\mathbf{L}_\gamma(\mathcal{P}') \succeq \mathbf{L}_\gamma(\mathcal{P}^{\text{tsp}})$ (Lemma 6 of [10]), and thus $\mathbf{L}_\gamma^{-1}(\mathcal{P}^{\text{tsp}}) \succeq \mathbf{L}_\gamma^{-1}(\mathcal{P}')$. It is obvious that $\gamma_{ij} \cdot \mathbf{B}_{ij}^\top \mathbf{L}_\gamma^{-1}(\mathcal{P}^{\text{tsp}}) \mathbf{B}_{ij} \geq \gamma_{ij} \cdot \mathbf{B}_{ij}^\top \mathbf{L}_\gamma^{-1}(\mathcal{P}') \mathbf{B}_{ij}$. Therefore,

$$(1 + \gamma_{ij} \cdot \mathbf{B}_{ij}^\top \mathbf{L}_\gamma^{-1}(\mathcal{P}^{\text{tsp}}) \mathbf{B}_{ij})^{\frac{1}{n}} \geq (1 + \gamma_{ij} \cdot \mathbf{B}_{ij}^\top \mathbf{L}_\gamma^{-1}(\mathcal{P}') \mathbf{B}_{ij})^{\frac{1}{n}}.$$

Considering $D(\mathcal{P}') \leq 2 \cdot D(\mathcal{P}^{\text{tsp}})$, it holds that $1 + \frac{2\omega(z_{ij})}{D(\mathcal{P}')^n} \geq 1 + \frac{\omega(z_{ij})}{D(\mathcal{P}^{\text{tsp}})}$. Combining the above two inequalities and Eq. (4), it holds that $\delta(z_{ij}, \mathcal{P}') < 1$, thus proves the Prop. 1.

With Prop. 1, we can further establish a distance threshold ω_{\max} to prune long loop edges from \mathcal{S} . We first find the loop edge $z_{ij}^* = \arg \max_{z_{ij}} (1 + \gamma_{ij} \cdot \mathbf{B}_{ij}^\top \mathbf{L}_\gamma^{-1}(\mathcal{P}^{\text{tsp}}) \mathbf{B}_{ij})^{\frac{1}{n}}$ with computational complexity $\mathcal{O}(|\mathcal{V}|^2)$. And then the distance threshold ω_{\max} satisfies $\frac{(1 + \gamma_{ij}^* \cdot \mathbf{B}_{ij}^{*\top} \mathbf{L}_\gamma^{-1}(\mathcal{P}^{\text{tsp}}) \mathbf{B}_{ij}^*)^{\frac{1}{n}}}{1 + \frac{\omega_{\max}}{D(\mathcal{P}^{\text{tsp}})}} = 1$ according to Prop. 1. If a loop edge $z_{ij} \in \mathcal{S}$ has distance $\omega(z_{ij}) > \omega_{\max}$, it will always satisfy Eq. (4) and thus can be removed from \mathcal{S} directly. Furthermore, we can also use \mathcal{P}' rather than \mathcal{P}^{tsp} in Eq. 4 to update the pruning threshold (Line 10, Alg. 1). The correctness can be proved similarly as in Prop. 1. The updated threshold is shown to be tighter and thus keeps pruning edges in greedy iterations, as in Fig. 4(c).

By applying Alg. 1, we can obtain a suboptimal set \mathcal{S}^* of selected loop edges. And then the final exploration path, denoted as \mathcal{P}^* , is constructed by inserting the loop edges in \mathcal{S}^* into their corresponding positions in \mathcal{P}^{tsp} following the insertion rule. The obtained path \mathcal{P}^* will provide high-level

³The assumption $D(\mathcal{P}') \leq 2 \cdot D(\mathcal{P}^{\text{tsp}})$ is reasonable because in the worst case, the robot can go back to the starting position by following exactly the initial path \mathcal{P}^{tsp} . The algorithm should find a better path than the worst case. The assumption is also verified to hold in our experiments.

Algorithm 1: GreedyLoopEdgeSelection

```

Input :  $\mathcal{G}^{\text{prior}}$ ,  $\mathcal{P}^{\text{tsp}}$ ,  $\mathcal{S}$ 
1 Find  $z_{ij}^*$  in  $\mathcal{S}$ , and calculate  $\omega_{\max}$ .
2  $\forall z_{ij} \in \mathcal{S}$ , if  $\omega(z_{ij}) \geq \omega_{\max}$ , remove  $z_{ij}$  from  $\mathcal{S}$ .
3  $\mathcal{S}' \leftarrow \{\}$ ,  $\text{iter} \leftarrow 0$ ,  $\mathcal{P}' \leftarrow \mathcal{P}^{\text{tsp}}$ .
4 while true do
5   for each  $z_{ij} \in \mathcal{S}$  do
6     Compute  $\delta(z_{ij}, \mathcal{P}')$ .
7     Update Eq. (4) using  $\mathcal{P}'$  rather than  $\mathcal{P}^{\text{tsp}}$ .
8     if  $z_{ij}$  satisfies Eq. (4) then
9       Remove  $z_{ij}$  from  $\mathcal{S}$ .
10    Find  $z_{ij}^{\max} \leftarrow \arg \max_{z_{ij} \in \mathcal{S}} \delta(z_{ij}, \mathcal{P}')$ .
11    if  $\delta(z_{ij}^{\max}, \mathcal{P}') > 1$  then
12      Add  $z_{ij}^{\max}$  into  $\mathcal{S}'$ , remove  $z_{ij}^{\max}$  from  $\mathcal{S}$ .
13      Update  $\mathcal{P}' \leftarrow \mathcal{P}^{\text{tsp}} \cup \mathcal{S}'$  following insertion rule.
14       $\text{iter} \leftarrow \text{iter} + 1$ .
15    else
16      Break.
17 return  $\mathcal{S}^* \leftarrow \mathcal{S}'$ .

```

guidance to the robot to explore the entire environment in a hierarchical framework, which will be introduced in Sec. V.

V. HIERARCHICAL EXPLORATION AND REPLANNING

This section presents a hierarchical exploration framework incorporating the proposed SLAM-aware path planner to achieve SLAM-aware exploration, as shown in Fig. 2.

A. Hierarchical Autonomous Exploration

As shown in Fig. 2, in our framework, the SLAM-aware path planner takes the prior topo-metric graph $\mathcal{G}^{\text{prior}}$ from users as input, and outputs the path \mathcal{P}^* as high-level guidance to the robot in exploration. The detailed steps are as follows:

- 1) The robot navigates to the first region (vertex) provided by the path \mathcal{P}^* ;
- 2) Active loop-closing: if current vertex is a loop-closing vertex, the robot will follow a segment of its previous trajectory to establish loop closures in current region.
- 3) Frontier exploration: otherwise, the robot exhaustively explores frontiers within current region (corresponds to a vertex in $\mathcal{G}^{\text{prior}}$);
- 4) The robot navigates to the next region (vertex) in \mathcal{P}^* .

The exploration strategy is robust to slight variations in the exact positions of vertices specified in $\mathcal{G}^{\text{prior}}$, because the robot is considered to have reached a vertex v_i once it enters the vicinity of v_i , without the need to exactly reach that vertex. Moreover, the robot can skip a non-loop closing vertex if the corresponding region has no frontiers. The resulting exploration trajectory is therefore mainly motivated by exploring the unknown regions, rather than exactly following the path \mathcal{P}^* , as shown in Fig. 1 and Fig. 3.

B. Online Prior Graph Update

This section introduces the online update of the prior graph as the robot obtains more information about the environment during exploration, to benefit the path planning process.

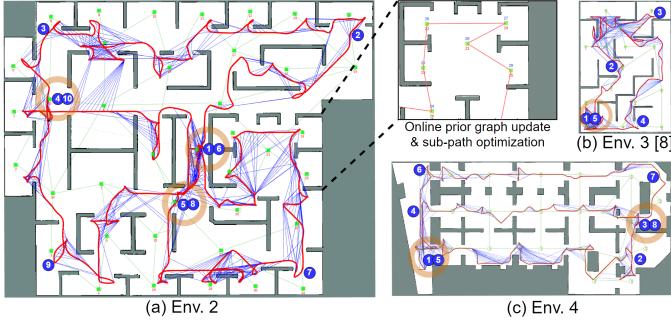


Fig. 3. The occupancy maps built by the proposed exploration method in various environments. The sizes of environment maps are: (a) $87m \times 69m$, (b) $40m \times 57m$, (c) $138m \times 66m$. The Env. 1 ($74m \times 74m$) is shown in Fig. 1. Each image also shows the robot's pose graph (red), loop closures (blue), the prior topo-metric graph (vertices and edges in green), and the ID of each vertex (numbers in red). The numbers in blue circles indicate the order in which the robot visits different regions, and the loop-closing vertices are marked by orange rings. Fig. 3(a) also shows a zoom-in area of Env. 2. The initial prior graph of Env. 2 only provides a star-like connectivity between the vertices. After the online connectivity update, the sub-path optimization module finds a better sub-path for exploration.

Degeneracy update: The environmental degeneracy can be reflected by the covariance of the relative motion estimation between two poses, e.g., the covariance in the scan-matching process [23], [24]. The robot is more likely to establish reliable loop closures in feature-rich regions that have low covariance in relative motion estimation. After a robot covers a region (vertex) v_i , the degeneracy matrix of this region is calculated by the averaged covariance matrix of the closest N edges in the robot's pose graph to vertex v_i (we set $N = 5$ in our experiments). For regions that have not been visited yet, their degeneracy matrices are defined as the averaged covariance matrix of all edges in the robot's pose graph. We then update the covariance matrix of an edge z_{ij} in the abstracted pose graph as the averaged degeneracy matrices of the two regions it connects.

Connectivity update: To supplement the potentially limited connectivity information in the initial prior graph, an independent thread will subscribe the SLAM-built map and run the A* algorithm to incrementally establish edges not specified in the prior graph.

C. Path Replanning and Sub-Path Optimization

After performing a loop-closing action, the framework will replan the robot's path by calling the SLAM-aware path planner based on the updated prior graph $\mathcal{G}^{\text{prior}}$. The robot will follow the better path of the existing path and the replanned path in subsequent exploration. Moreover, once the prior graph is updated, the sub-path of \mathcal{P}^* starting from the current vertex (where the robot is located) until the next loop-closing vertex is also optimized by solving a local TSP problem, trying to find a shorter sub-path based on the updated prior graph while not affecting the loop-closing behavior of the robot. An example is shown in Fig. 3(a).

VI. EXPERIMENTS

We implement the proposed algorithm as an exploration plugin to the *nav2d*⁴ ROS package. We use *pyconcorde*⁵ as the TSP solver. The robot trajectory error is evaluated using *evo*⁶. All experiments are conducted on a desktop with an i9-13900 CPU and 32 GB of RAM. The experimental environments Env. 1-4 and the associated prior topo-metric graphs are shown in Fig. 1 and Fig. 3. We compare our method with the frontier-based method [25], TSP-based method [8] (not open-source and we implement our version), and the active graph-based SLAM (AGS) method in [7]. Note the proposed method and the TSP-based method require prior topo-metric information, while others do not have such constraints. All these implementations are open-sourced.⁷

A. Efficiency of Pruning Threshold in Loop Edge Selection

This section evaluates the effectiveness of the pruning thresholds in Alg. 1, based on randomly generated prior graphs with different areas, i.e., $10m \times 10m$, $15m \times 15m$, $20m \times 20m$, $30m \times 30m$, as shown in Fig. 4(a). The graphs are derived from grid-like structures, with 5% of vertices and edges randomly removed to introduce diverse topology. We also add the Gaussian noise with zero mean and $\sigma = 0.2m$ to the positions of vertices in prior graphs. The edges in the prior graphs are assumed to have constant measurement covariance $\text{diag}\{0.1m, 0.1m, 0.001rad\}$.

Tab. I records the number of candidate edges before and after pruning in the first iteration of Alg. 1, where the distance threshold ω_{\max} prunes around half of the candidate edges in most cases, and Prop. 1 further prunes over 90% of the candidate edges in \mathcal{S} . The pruning ratio gets smaller as the size of the graph increases, indicating the proposed thresholds work better in large-scale environments or more fine-grained prior topo-metric graphs. Moreover, the runtime efficiency of Alg. 1 with and without the proposed pruning thresholds is also shown in Tab. I, where $t_{\text{prune}}^{\text{Alg.1}}$ is almost an order of magnitude faster than $t_{\text{no-prune}}^{\text{Alg.1}}$. Fig. 4(b) shows the proportion of remaining loop edges during the iteration of Alg. 1 using the updated pruning threshold Eq. (4) in Prop. 1, as introduced in Sec. IV-D. In all cases, the updated threshold in Prop. 1 prunes a significant number of candidate loop edges in the first several iterations, indicating the updated threshold gets even tighter as Alg. 1 iterates.

In addition, we also record the time spent to solve the TSP problem over the example prior graphs in Tab. I. Two methods are compared, a branch-and-bound (BnB) method using *pyconcorde* and a greedy-based method⁸. Although the BnB method usually outputs better TSP paths, it becomes inefficient when dealing with hundreds of vertices, in which case the greedy method can be used to get a valid sub-optimal TSP path within a second. In the exploration of Env. 1-4, we use the *pyconcorde* package which can output the TSP path within 0.1 seconds.

⁴https://github.com/skasperski/navigation_2d

⁵<https://github.com/jvkersch/pyconcorde>

⁶<https://github.com/MichaelGrupp/evo>

⁷https://github.com/bairuofei/Graph-Based_SLAM-Aware_Exploration

⁸<https://github.com/dmishin/tsp-solver>

TABLE I
EFFECTIVENESS OF PRUNING THRESHOLDS IN ALG. 1

Size (m^2)	10×10	15×15	20×20	30×30
$ \mathcal{V} $	95	220	395	895
$ \mathcal{S} $	4,371	22,578	71,630	364,214
Aft. ω_{\max}	4,058	13,295	29,008	69,864
Aft. Prop. 1	329	1,272	2,190	4,743
Ratio	7.53%	5.63%	3.06%	1.63%
$t_{\text{BnB}}^{\text{TSP}}$ (sec)	0.41	1.28	42.15	-
t_{greedy} (sec)	0.004	0.02	0.08	0.44
$t_{\text{Alg.1}}^{\text{no-prune}}$ (sec)	0.48	5.53	50.21	1181.61
$t_{\text{Alg.1}}^{\text{prune}}$ (sec)	0.07	0.59	3.01	21.47

Note: “-” means cannot return a solution after 30 minutes; $t_{\text{no-prune}}^{\text{Alg.1}}$ and $t_{\text{prune}}^{\text{Alg.1}}$, the runtime of Alg. 1 w/wo pruning thresholds; $t_{\text{BnB}}^{\text{TSP}}$ and $t_{\text{greedy}}^{\text{TSP}}$, the TSP solving time using a branch-and-bound method and a greedy method.

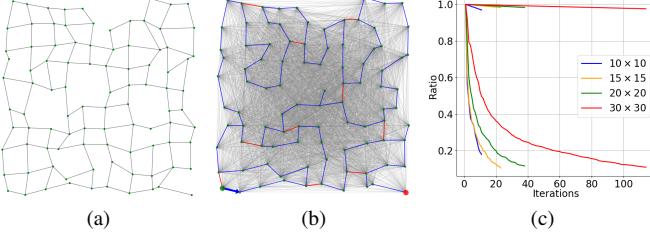


Fig. 4. (a) An example topo-metric graph derived from a $10m \times 10m$ grid-like structure; (b) the initial TSP walk P^{TSP} (blue), the candidate (gray) and the selected (red) loop edges. (c) The ratio of the remaining candidate loop edges after the first iteration of Alg. 1 until termination. The curves in the upper area of (c) show the corresponding cases without edge pruning.

B. Evaluation of Exploration Performance

This section evaluates the exploration performance of our method in Env. 1-4, as shown in Fig. 1 and Fig. 3. All methods run the same SLAM method called *Karto* [24] with both odometry estimation and loop closure detection.

Tab. II records the averaged exploration results over five independent runs. In all cases, our method has the smallest absolute pose error, and a comparable exploration efficiency (quantitated by the total distance d_{total}) with the frontier-based method except in Env. 1, which is close to the free space and thus our method needs additional movement for active loop closing. The TSP-based method covers the whole environment with the shortest distance; however, this comes at the cost of larger pose errors. The active SLAM method [7] gets better exploration efficiency in Env. 3 than our methods, and good mapping accuracy in Env. 4, because it evaluates the information gain and pose graph reliability when selecting frontiers. However, as it is mainly motivated by exploring frontiers, the method fails to actively form informative loop closures to reduce the odometry error. It is also verified in Tab. II that a larger averaged node degree k (i.e., more loop closures) does not always lead to a smaller pose error, while only the globally informative ones can effectively mitigate the accumulated odometry drift in exploration.

Fig. 5 shows the cumulative density distributions of the pose uncertainty in the robot’s pose graph. The curve that converges faster to one indicates a higher proportion of poses have smaller estimation uncertainty. Generally, the curves of our method converge faster to one than other methods, with obviously smaller variances among different runs. In comparison, both the frontier-based and TSP-based methods show larger variations and converge slower. The method [7] shows a comparable uncertainty distribution to ours in Env. 1,

TABLE II
EXPLORATION RESULTS IN DIFFERENT ENVIRONMENTS

Env.	Method	n_{pose}	k	APE / m	$t_{\text{total}}/\text{sec}$	$d_{\text{total}}/\text{m}$
1	Fron. [25]	532 \pm 17	3.02	0.28 \pm 0.13	932.72	405.42
	TSP [8]	512 \pm 30	2.91	0.41 \pm 0.35	932.18	385.02
	AGS [7]	821 \pm 64	3.34	0.33 \pm 0.09	4077.06	627.75
	Ours	625 \pm 19	3.02	0.20 \pm 0.07	1262.58	488.69
2	Frontier	1062 \pm 63	3.86	0.21 \pm 0.14	1428.56	747.07
	TSP	922 \pm 29	3.46	0.23 \pm 0.24	1287.14	689.40
	AGS	1499 \pm 274	5.51	0.28 \pm 0.01	6568.32	989.66
	Ours	1060 \pm 58	3.85	0.19 \pm 0.10	1418.04	783.50
3	Frontier	458 \pm 49	3.86	0.14 \pm 0.09	619.58	349.60
	TSP	343 \pm 52	3.74	0.14 \pm 0.12	486.38	276.06
	AGS	377 \pm 35	3.45	0.21 \pm 0.02	1255.16	304.92
	Ours	437 \pm 43	4.27	0.11 \pm 0.06	642.0	348.67
4	Frontier	1093 \pm 148	3.82	0.30 \pm 0.18	1669.66	804.272
	TSP	849 \pm 58	3.47	0.78 \pm 0.62	1223.66	637.89
	AGS	1013 \pm 67	3.47	0.29 \pm 0.05	4790.3	848.31
	Ours	947 \pm 53	3.66	0.24 \pm 0.13	1710.18	748.88

Note: n_{pose} , the number of poses in pose graph; $k = (2 \times n_{\text{edge}})/n_{\text{pose}}$, the averaged node degree in pose graph; APE, the RMSE of absolute pose error; t_{total} and d_{total} , the total time and distance used to cover the environment.

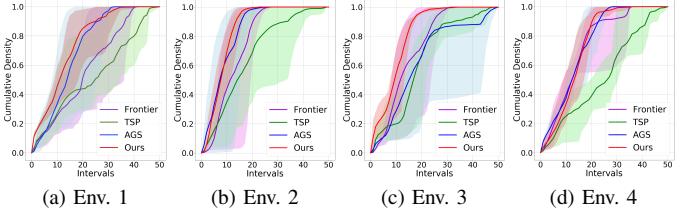


Fig. 5. Cumulative density distribution of the robot poses uncertainty. Each curve (with the lower and upper bound) is the averaged result over five independent runs. The pose uncertainty is evaluated by the determinant of the covariance matrix associated with each pose (using GTSAM [26]) in the pose graph, and the distribution of pose uncertainty is discretized into 50 sub-intervals to compute its cumulative density function.

2, and 4, at the cost of longer distances that introduces extra odometry noise to the pose estimation as in Tab. II.

Moreover, we compared the waypoint planning time during the exploration of Env. 1, as shown in Fig. 6(a). The waypoints are low-level navigation goals decided after evaluating existing frontiers or loop-closing opportunities, and thus the planning time reflects the online decision-making efficiency of the compared methods. As the active SLAM method [7] needs to construct a local pose graph when evaluating each frontier, it usually takes tens of seconds to find the best one and thus is computationally inefficient. Our method has much better runtime efficiency than [7] because we only evaluate the pose graph reliability with its coarse structure. The runtime efficiency of our method is similar to the TSP-based method, but with additional capabilities of online prior graph updating and path replanning. Moreover, we evaluate the performance of the proposed method using the prior graph of Env. 1 with different levels of noise, as shown in Fig. 6(b). The averaged total exploration distance and the absolute pose error are less affected, verifying our exploration strategy is robust to the slight variations in the prior graph, as explained in Sec. V-A.

C. Real-World Experiment with Pioneer3AT Robot

We validated the proposed method by exploring a structured corridor environment on the NTU campus, as shown in Fig. 7. The topo-metric information is obtained from the satellite map, and about half of the vertices in the prior graph are wrongly placed within obstacles. The proposed method uses such an inaccurate prior graph, finds the path that quickly covers the environment and includes three active loop-closing actions to mitigate the odometry drift during exploration. The obtained map is shown in Fig. 7 (b). More details are included in the attached video.

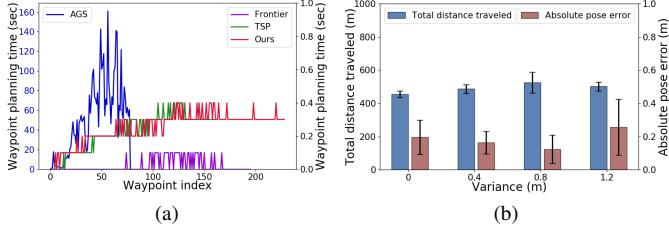


Fig. 6. (a) A record of the waypoint planning time during the exploration of Env. 1. The left y-axis only corresponds to the AGS method, while the right y-axis corresponds to the other methods. (b) The total exploration distance and the RMSE of absolute pose error of the proposed method when Gaussian noise with variance $0.4m$, $0.8m$ and $1.2m$ are added to the positions of vertices in the prior graph of Env. 1. The averaged results and standard deviations over five independent runs are shown.

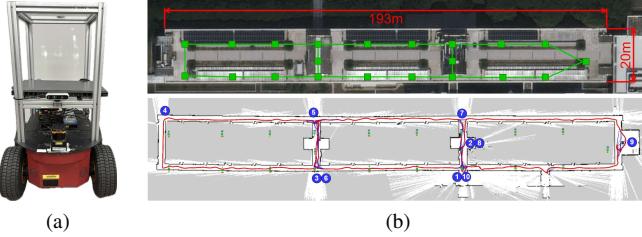


Fig. 7. (a) The Pioneer3AT robot equipped with wheel encoders, a Hokuyo UTM-30LX laser, and Intel RealSense D435 camera (for visualization only); (b) Top: topview of the building from satellite image; Bottom: the SLAM-built map after exploration and the robot pose graph.

VII. CONCLUSION

This letter proposes a SLAM-Aware path planner for autonomous exploration based on a prior topo-metric graph of the environment, which finds a high-level path to guide the robot to quickly cover unknown regions while forming a reliable SLAM pose graph with informative loop closures. A hierarchical exploration framework is designed to incorporate the proposed planner, with additional features including online prior graph update and path replanning. Experimental results verify that the proposed method achieves a superior balance between mapping accuracy and exploration efficiency in various environments. The proposed method is specifically applicable for exploring environments with rich topology information, like industrial factories, architectural complexes, or underground mines, where the topo-metric information can be effectively exploited. Future work is to construct the topo-metric graph online and extend it to the multi-robot case.

REFERENCES

- [1] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard, "A tutorial on graph-based SLAM," *IEEE Intelligent Transportation Systems Magazine*, vol. 2, no. 4, pp. 31–43, 2010.
- [2] H. Guo, J. Zhu, and Y. Chen, "E-loam: Lidar odometry and mapping with expanded local structural information," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 2, pp. 1911–1921, 2023.
- [3] J. A. Placed and J. Strader, H. Carrillo, N. Atanasov, V. Indelman, L. Carbone, and J. A. Castellanos, "A survey on active simultaneous localization and mapping: State of the art and new frontiers," *IEEE Transactions on Robotics (TRO)*, vol. 39, no. 3, pp. 1686–1705, 2023.
- [4] C. Stachniss, G. Grisetti, and W. Burgard, "Information gain-based exploration using rao-blackwellized particle filters," in *Robotics: Science and Systems (RSS)*, vol. 2, pp. 65–72, 2005.
- [5] Y. Xu, R. Zheng, M. Liu, and S. Zhang, "Crmi: Confidence-rich mutual information for information-theoretic mapping," *IEEE Robotics and Automation Letters (RA-L)*, vol. 6, no. 4, pp. 6434–6441, 2021.
- [6] H. Carrillo *et al.*, "Autonomous robotic exploration using a utility function based on Rényi's general theory of entropy," *Autonomous Robots*, vol. 42, no. 2, pp. 235–256, 2018.
- [7] J. A. Placed and J. A. Castellanos, "Fast autonomous robotic exploration using the underlying graph structure," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6672–6679, 2021.
- [8] S. Oßwald *et al.*, "Speeding-up robot exploration by exploiting background information," *IEEE Robotics and Automation Letters (RA-L)*, vol. 1, no. 2, pp. 716–723, 2016.
- [9] W. Xue, R. Ying, F. Wen, Y. Chen, and P. Liu, "Active SLAM with prior topo-metric graph starting at uncertain position," *IEEE Robotics and Automation Letters (RA-L)*, vol. 7, no. 2, pp. 1134–1141, 2022.
- [10] K. Khosoussi, M. Giamou, G. S. Sukhatme, S. Huang, G. Dissanayake, and J. P. How, "Reliable graphs for SLAM," *The International Journal of Robotics Research (IJRR)*, vol. 38, no. 2-3, pp. 260–298, 2019.
- [11] Y. Chen, S. Huang, L. Zhao, and G. Dissanayake, "Cramér–Rao bounds and optimal design metrics for pose-graph SLAM," *IEEE Transactions on Robotics (TRO)*, vol. 37, no. 2, pp. 627–641, 2021.
- [12] J. A. Placed and J. A. Castellanos, "A general relationship between optimality criteria and connectivity indices for active graph-SLAM," *IEEE Robotics and Automation Letters (RA-L)*, vol. 8, no. 2, pp. 816–823, 2023.
- [13] M. Luperto, M. Antonazzi, F. Amigoni, and N. A. Borghese, "Robot exploration of indoor environments using incomplete and inaccurate prior knowledge," *Robotics and Autonomous Systems (RAS)*, vol. 133, p. 103622, 2020.
- [14] A. Soragna, M. Baldini, D. Joho, R. Kümmerle, and G. Grisetti, "Active SLAM using connectivity graphs as priors," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 340–346, 2019.
- [15] M. Deshpande, R. Kim, D. Kumar, J. J. Park, and J. Zamiska, "Light-houses and global graph stabilization: Active slam for low-compute, narrow-fov robots," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4070–4076, 2023.
- [16] K. Khosoussi *et al.*, "Novel insights into the impact of graph structure on SLAM," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2707–2714, 2014.
- [17] Y. Chen, S. Huang, R. Fitch, L. Zhao, H. Yu, and D. Yang, "Online 3D active pose-graph SLAM based on key poses using graph topology and sub-maps," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 169–175, 2019.
- [18] H. Carrillo *et al.*, "On the comparison of uncertainty criteria for active SLAM," in *2012 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2080–2087, 2012.
- [19] A. Pázman, "Foundations of optimum experimental design," 1986.
- [20] M. Jünger *et al.*, "Chapter 4 the traveling salesman problem," in *Network Models*, vol. 7 of *Handbooks in Operations Research and Management Science*, pp. 225–330, Elsevier, 1995.
- [21] H. H. Chieng and N. Wahid, "A performance comparison of genetic algorithm's mutation operators in n-cities open loop travelling salesman problem," in *Proceedings of The First International Conference on Soft Computing and Data Mining*, pp. 89–97, Springer, 2014.
- [22] G. Grisetti, R. Kümmerle, C. Stachniss, U. Frese, and C. Hertzberg, "Hierarchical optimization on manifolds for online 2d and 3d mapping," in *2010 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 273–278, 2010.
- [23] J. Zhang, M. Kaess, and S. Singh, "On degeneracy of optimization-based state estimation problems," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 809–816, 2016.
- [24] K. Konolige, G. Grisetti, R. Kümmerle, W. Burgard, B. Limketkai, and R. Vincent, "Efficient sparse pose adjustment for 2d mapping," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 22–29, 2010.
- [25] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, pp. 146–151, 1997.
- [26] F. Dellaert and G. Contributors, "borglab/gtsam," May 2022.