IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

ERPoT: Effective and Reliable Pose Tracking for Mobile Robots Using Lightweight Polygon Maps

Haiming Gao, Qibo Qiu, Hongyan Liu, Dingkun Liang, Chaoqun Wang, Xuebo Zhang

***Abstract*— This paper presents an effective and reliable pose tracking solution, termed ERPoT, for mobile robots operating in large-scale outdoor and challenging indoor environments, underpinned by an innovative prior polygon map. Especially, to overcome the challenge that arises as the map size grows with the expansion of the environment, the novel form of a prior map composed of multiple polygons is proposed. Benefiting from the use of polygons to concisely and accurately depict environmental occupancy, the prior polygon map achieves long-term reliable pose tracking while ensuring a compact form. More importantly, pose tracking is carried out under pure LiDAR mode, and the dense 3D point cloud is transformed into a sparse 2D scan through ground removal and obstacle selection. On this basis, a novel cost function for pose estimation through point-polygon matching is introduced, encompassing two distinct constraint forms: point-to-vertex and point-to-edge. In this study, our primary focus lies on two crucial aspects: lightweight and compact prior map construction, as well as effective and reliable robot pose tracking. Both aspects serve as the foundational pillars for future navigation across diverse mobile platforms equipped with different LiDAR sensors in varied environments. Comparative experiments based on the publicly available datasets and our self-recorded datasets are conducted, and evaluation results show the superior performance of ERPoT on reliability, prior map size, pose estimation error, and runtime over the other six approaches. The corresponding code can be accessed at `https://github.com/ghm0819/ERPoT`, and the supplementary video is at `https://youtu.be/6XdcXyUrLKw`.**

I. INTRODUCTION

Mobile robotics is the trending area where related research has made great progress in the past decade, with accurate localization and robust navigation emerging as cornerstones for the development of autonomous capabilities [1], [2]. As the application of mobile robots extends into more complex environments, the demand for high-precision and reliable localization systems intensifies [3].

This work was supported in part by the National Natural Science Foundation of China under Grant 62303428, and in part by Beijing-Tianjin-Hebei Fundamental Research Cooperation Project under Grant 24JCZXJC00390. *(Corresponding author: Dingkun Liang)*

Haiming Gao is with the ZJU-Hangzhou Global Scientific and Technological Innovation Center, Zhejiang University, Hangzhou, P. R. China, 311215 (Email: ghm@zju.edu.cn).

Qibo Qiu is with State Key Lab of CAD&CG, Zhejiang University, and also with China Mobile (Zhejiang) Research & Innovation Institute.

Hongyan Liu is with Beijing Fuyouhua Intelligent Technology Co., Ltd.

Dingkun Liang is with College of Information Engineering, Zhejiang University of Technology, Hangzhou, P. R. China, 310023 (Email: liangdk@zjut.edu.cn).

Chaoqun Wang is with School of Control Science and Engineering, Shandong University, Jinan, P. R. China, 250061.

Xuebo Zhang is with the Institute of Robotics and Automatic Information System (IRAIS), Tianjin Key Laboratory of Intelligent Robotics (TJKLIR), Nankai University, Tianjin, P. R. China, 300350.

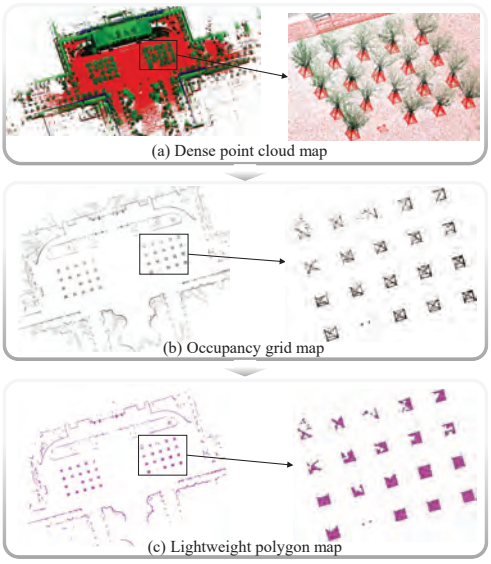


Fig. 1. Lightweight and compact polygon map construction.

In response, many state-of-the-art approaches [4]–[6] have been proposed to obtain environmental maps, which serve as the prior information for robust localization of mobile robots in various applications such as service [7], navigation [8], and exploration [9]. When both the prior map and the initial pose are provided, the localization problem can be regarded as *pose tracking* [10], which continuously estimates the robot pose. The pursuit of effective and reliable approaches for pose tracking is central to advancing the field of mobile robotics.

More importantly, the effectiveness and reliability of pose tracking fundamentally depend on the quality of the prior map used for organizing environmental elements, and the superiority of the algorithms employed for processing sensor data and achieving accurate data association. For pose tracking methodologies that leverage LiDAR sensors, they can generally be categorized into key approaches, including point cloud registration techniques [11]–[13], feature-based matching [14]–[16], machine learning-based algorithms [6], [17], [18], and various filter-based methods [19]–[21]. These diverse strategies reflect the multifaceted nature of pose estimation within the field of autonomous systems. However, despite the success of previous pose tracking methods, they often face certain challenges, such as the large size of prior maps [22], particularly in large-scale environments, which can hinder computational efficiency and map storage. In addition, complex pose tracking processes with specific

semantic clues and structure characteristics may not always guarantee effectiveness and reliability [23]. In light of these challenges, there is a pressing need for innovative solutions that can offer both *compact map form* and *reliable localization technique*.

Motivated by the requirements mentioned above, we propose an effective and reliable pose tracking approach for mobile robots based on the novel prior map, called ERPoT. For mobile robots operating in terrestrial environments, the roll and pitch angles are typically stable and exhibit minimal variation [24]. Consequently, the proposed ERPoT, which operates on three degrees of freedom, is generally adequate to address the requirements of nearly all task scenarios effectively. In addition, the proposed pose tracking framework is a general solution that applies to various environments and is not limited to environments containing specific semantic elements. Specifically, LiDAR data serves as the sole input, from which a sparse 2D scan is derived through ground removal and obstacle selection. Subsequently, an offline map construction process is performed to acquire a lightweight and compact polygon map of the environment, as shown in Fig. 1. Multi-polygon-based representation (Fig. 1(c)) offers superior advantages over point cloud (Fig. 1(a)) and occupancy grid (Fig. 1(b)), particularly in applications demanding precision and efficient data utilization [25]. Finally, we can make use of the 2D scan and the prior polygon map to realize effective and reliable pose tracking for mobile robots, which takes the advantage of the newly proposed point-polygon matching. This research focuses on constructing lightweight and compact prior maps and establishing effective and reliable pose tracking. The main contributions include three aspects:

(1) **Innovative form of the prior map**: The prior map composed of multiple polygons is proposed for pose tracking of mobile robots, which effectively compresses the environmental information without relying on any semantic elements. More importantly, the prior polygon map realizes the lightweight and compact representation of the large-scale environment, while ensuring effective and reliable pose tracking performance.

(2) **Point-polygon matching-based pose tracking**: Based on the prior map, an effective and reliable pose tracking approach is proposed, called ERPoT. Firstly, the dense point cloud is compressed into a sparse 2D scan through ground removal and obstacle selection. On this basis, the newly proposed cost function on *point-polygon matching* is constructed, which includes *point-to-vertex* and *point-to-edge*.

(3) **Generalization across platforms and environments**: Extensive comparative experiments on public datasets (including building squares, road scenes, vegetation, *etc.*) and self-recorded datasets (including long-term changes, and terrain changes), which are acquired through different platforms equipped with different LiDAR sensors, demonstrating the effectiveness and reliability of our proposed approach.

The remaining parts of this paper are structured as follows. Section II describes related works on prior map-based pose tracking. Section III presents the effective and reliable pose tracking approach called ERPoT in detail. Section IV provides the experimental results to show the effectiveness of ERPoT. Finally, the paper is concluded in Section V.

II. Related work

Reliable pose tracking is the backbone of autonomous mobile robots, enabling them to move intelligently, interact safely, and perform complex tasks. In this section, we briefly discuss previous related works in the fields of LiDAR odometry, prior map construction, and pose tracking.

A. LiDAR odometry

LiDAR odometry, a critical technique for mobile robots, leverages LiDAR sensors to estimate the trajectory and construct a map of the environment in real-time. Current LiDAR odometry pipelines commonly employ an iterative closest point (ICP) algorithm and its variants to realize pose estimation incrementally. Especially, the authors in [11] present a voxelized generalized iterative closest point (VGICP) approach, bridging the gap between the generalized iterative closest point (GICP) and the normal distributions transform (NDT), which achieves fast and accurate 3D point cloud registration. By introducing the definition of continuous-time trajectory, Dellenbach *et al.* [12] propose a novel real-time LiDAR-only odometry approach called CT-ICP, combined with a novel loop detection procedure to realize a complete SLAM system. In recent years, research on ICP-based LiDAR odometry has been an active field. Vizzo *et al.* [13] present a novel pose estimation system called KISS-ICP, which is a lightweight and efficient ICP algorithm designed for real-time applications, emphasizing simplicity and performance without compromising accuracy. In addition, MAP-ICP is proposed in [26], utilizing an efficient and versatile KD-tree data structure and dynamically maintaining a robust environment model through the incorporation of estimated pose uncertainty.

However, GICP and other ICP-variants, which heavily depend on nearest neighbor search, can struggle to achieve real-time performance in large-scale environments, especially when operating on computers with limited computational capabilities [27]. On the other hand, feature-based registration methods first extract relatively fewer salient features from point clouds and then estimate the transformation only with the feature points. As the milestone work, LiDAR odometry and mapping (LOAM) is proposed in [14], computing the robot pose with planar and edge feature points. Inspired by the well-known LOAM, Shan *et al.* [15] propose the novel LeGO-LOAM, which adds ground constraints to improve the accuracy of localization and mapping. Furthermore, the authors in [5] have introduced a framework termed LIO-SAM, which excels in delivering highly accurate, real-time trajectory estimation and map-building capabilities for mobile robots. Recently, F-LOAM [28], presented as a comprehensive solution, aims to provide a computationally efficient and precise framework for LiDAR-based SLAM. In addition, to adapt to resource-limited platforms, a lightweight LiDAR

SLAM system, Light-LOAM [29], is proposed with comparative performance to the state-of-the-art methods, while also striking a balance with its computational requirements.

Different from the above classical LiDAR odometry systems, deep learning-based LiDAR odometry does not rely on hand-tuned feature extraction and data association, and has attracted much attention in the last few years. Especially, Neural Radiance Field (NeRF) presents an innovative approach to map representation, showing great potential for utilization in robotics applications. On this basis, Isaacson *et al.* [17] propose LONER, the first real-time neural implicit LiDAR SLAM adapting to outdoor environments with accurate online state estimation. To address the problem of NeRF-based LiDAR SLAM used in outdoor large-scale environments, the work in [6] proposes a novel NeRF-based LiDAR odometry and mapping approach, NeRF-LOAM, which obtains both dense 3D representation and accurate poses. On the other hand, several related works [18], [30] firstly extract local robust descriptors of the point cloud and then make use of classical refinement techniques to realize pose estimation.

B. Prior map construction

The method and form of prior map construction are fundamental to the performance of prior map-based pose tracking for mobile robots. PoseMap proposed in [31] integrates the trajectory directly into the prior map, each robot pose is associated with a submap consisting of surfels, transcending the limitations of purely metric map representations and enhancing the overall efficiency. This topological structure of prior maps can improve pose tracking efficiency in large-scale environments. To fulfill the increasing computational requirements that come with the size of the prior map, Feng *et al.* [32] propose a localization system based on Block Maps to reduce the computational load. In [33], Xie *et al.* present a solution for robust indoor localization in environments like offices and corridors, utilizing 3D LiDAR point clouds within a compact, hierarchical, topometric semantic map.

In contrast, leveraging salient objects and distinctive feature information within the environment allows for a more compact representation, thereby enhancing the robustness of pose estimation. Especially, the work in [22] proposes Segmap, a segment-based approach for map representation in localization and mapping based on the previous works [34], [35], enhancing traditional approaches with its distinctive, learning-based descriptors that offer superior descriptive power and competitive localization performance. On the other hand, pole-like objects are well suited to serve as landmarks for vehicle localization in urban environments due to their widespread presence and consistent reliability over time, Schaefer *et al.* [36] present a long-term 2D vehicle localization approach in urban environments with the extracted pole landmarks from mobile LiDAR data. Furthermore, the study in [16] capitalizes on extracted poles to train a deep neural network, facilitating online range image-based pole segmentation. This approach avoids processing the 3D point cloud, thereby enabling rapid pole extraction for long-term localization. Recently, based on the semantics of pole-like structures, the authors in [23] introduce a multi-layer semantic pole-map by aggregating the detected pole-like structures, and the semantic particle-filtering localization scheme is also proposed for vehicle localization.

Prior map construction methods mentioned above, when confronted with large-scale outdoor environments, employ special map structures like topology to enhance the efficiency of data association, yet struggle to effectively manage map size. On the other hand, leveraging semantically salient objects allows for a compact environmental description and ensures reliable pose tracking. However, this approach is only viable in outdoor settings that meet specific semantic requirements.

C. Pose tracking

Accurate prior map-based pose tracking allows mobile robots to determine their exact location within an environment, which is essential for navigating and interacting with their surroundings effectively. Especially, for autonomous navigation, self-driving cars commonly rely on precise localization within a prior map to guide their way [37]. In [38], Wolcott *et al.* model the world as a mixture of several Gaussians and present a generic probabilistic-based localization system that can handle harsh weather conditions and poorly textured roadways. Based on the mesh-based prior map, the authors in [19] introduce an innovative observation model, seamlessly integrating it within a Monte Carlo localization framework to enhance accuracy and robustness. The Monte Carlo-based localization method has been widely used in large-scale outdoor tasks due to its advantages in dealing with uncertainty, nonlinear problems, and computational efficiency. For instance, Akai [20] merges the strengths of Monte Carlo localization with those of scan matching, culminating in the proposal of an efficient map-based localization solution that capitalizes on the best of both methodologies. Considering non-flat terrain characteristics of the operational environments, Rico *et al.* [21] present a mobile robot localization algorithm grounded in AMCL, utilizing elevation maps and probabilistic 3D occupancy maps to determine the robot pose accurately.

On the other hand, scan matching also plays a pivotal role in mobile robot pose tracking by aligning sequential point cloud data with the prior map to compute the robot pose. In [39], Koide *et al.* propose a real-time sensor localization system, integrating the NDT scan matching [11] with an angular velocity-based pose prediction using an unscented Kalman filter (UKF). For long-term applications of mobile robots, the work in [40] introduces a resilient long-term LiDAR-based localization system, which incorporates a temporary mapping module to circumvent potential localization failures due to incorrect global matching. In contrast, in a data-driven manner, Hroob *et al.* [41] propose a novel stability scan filter for robust localization of field mobile robots through long-term stability landmarks in a continuously changing environment. In addition, the authors

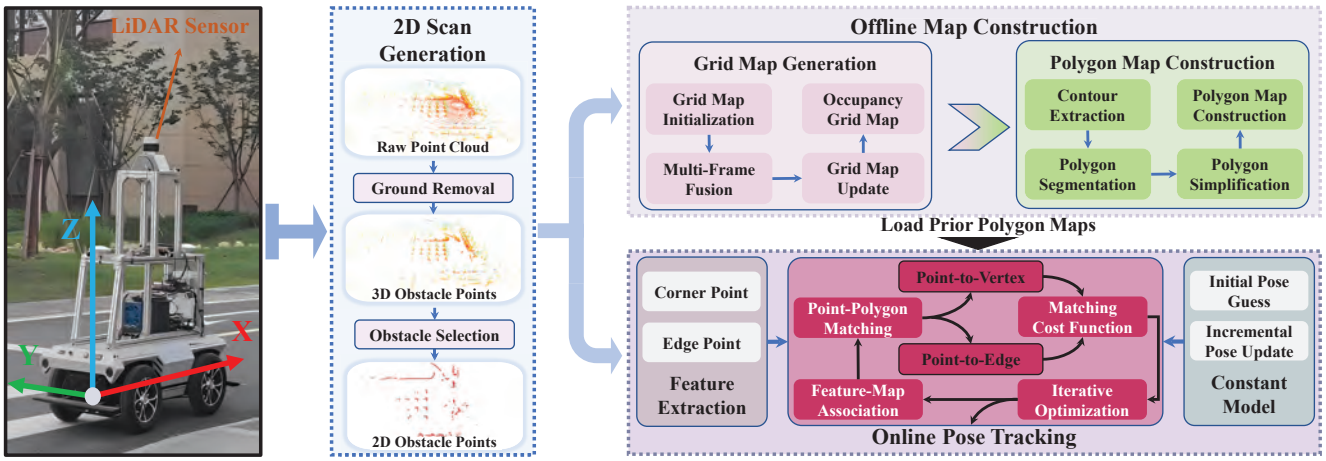


Fig. 2. The process of **E**ffective and **R**eliable **Po**se **T**racking approach (**ERPoT**) for mobile robots, which mainly includes two parts: offline map construction and online pose tracking. The left subfigure shows the experimental platform for collecting the self-recorded datasets, and the middle subfigure represents the point cloud preprocessing process, named 2D scan generation (**Section III-B**), which is used in the following two main processes. In particular, pink module and purple module represent offline map construction (**Section III-C**) and online pose tracking (**Section III-D**), respectively.

in [42] present a novel tightly-coupled graph localization approach that exploits prior topo-metric knowledge of the environment and seamlessly integrates various modules into a graph optimization framework.

For the aforementioned prior map-based pose tracking methods, a challenge arises as the map size grows with the expansion of the environment. Particularly in large-scale outdoor settings, the storage of prior maps can demand substantial memory resources, and the large maps can present significant computational challenges for pose tracking. This challenge is the driving force behind our novel polygon map approach, which ensures minimal map sizes in extensive environments without compromising the reliability of pose tracking.

III. Effective and Reliable Pose Tracking

The detailed procedures (Fig. 2) of the proposed ERPoT are introduced in this section. At first, the LiDAR data is the only input information, and the corresponding sparse 2D scan is acquired through the point cloud preprocessing process. Afterward, the process of offline map construction is carried out, and the lightweight and compact polygon map of the environment can be obtained. Based on the above process, we can make use of the real-time 2D scan and the polygon map to realize effective and reliable pose tracking.

A. Problem formulation

The problem of pose tracking can be formulated as two main parts: offline map construction and online pose tracking.

1) Offline map construction: We propose a novel form of map representation called polygon map, which is more lightweight and more compact compared with traditional dense point cloud maps. To be specific, the input sensor data only includes the LiDAR data, which is represented by $\mathcal{L}_t = \{\mathbf{l}_1, \mathbf{l}_2, ..., \mathbf{l}_t\}$ with t being the timestamp, while the i-th frame $\mathbf{l}_i$ is denoted by the set $\{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_n\}$ with n being the number of points, and $\mathbf{p}_i = [x_i, y_i, z_i]$.

On the other hand, the corresponding pose sequence $\mathcal{S}_t = \{\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_t\}$ is also required in map construction, and $\mathbf{s}_i = [x_i, y_i, \theta_i]$ with θ_i being the yaw angle. Note that $\mathcal{S}_t$ can be obtained through several LiDAR SLAM approaches or the provided ground truth from public datasets. More importantly, the polygon map is denoted by $\mathcal{M}_p$, which is composed of multiple polygons $\{\mathbf{P}_1, \mathbf{P}_2, ..., \mathbf{P}_m\}$ with m being the number of polygons. Therefore the joint posterior probability density function by time step t is defined as

$$p\left(\mathcal{M}_p|\mathcal{L}_t, \mathcal{S}_t\right). \tag{1}$$

In practical applications, we cannot directly obtain the polygon map $\mathcal{M}_p$ from LiDAR data sequence $\mathcal{L}_t$ and pose sequence $\mathcal{S}_t$. Accordingly, let the occupancy grid map denoted as $\mathcal{M}_g$ as the intermediate form, and equation (1) can be rewritten as

$$p\left(\mathcal{M}_p|\mathcal{L}_t, \mathcal{S}_t\right) = \int_{\mathcal{M}_g} p\left(\mathcal{M}_p|\mathcal{M}_g\right) \cdot p\left(\mathcal{M}_g|\mathcal{L}_t, \mathcal{S}_t\right), \tag{2}$$

where the occupancy grid map $\mathcal{M}_g$ is composed of independent grids with fixed size, the process of grid map generation can refer to Karto-SLAM [43], while $p\left(\mathcal{M}_p|\mathcal{M}_g\right)$ represents the process of polygon generation. Before generating the 2D occupancy grid map, we need to convert 3D LiDAR points into 2D scan information via ground removal and obstacle selection, as shown in the middle subfigure in Fig. 2.

2) Online pose tracking: Given the prior polygon map of the environment and the initial pose $\tilde{\mathbf{s}}_1$, then the process of pose tracking can be carried out. Specifically, the robot pose at timestamp t is denoted by $\mathbf{s}_t$, and pose tracking result is denoted by the pose sequence $\mathcal{S}_t = \{\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_t\}$. In addition, let $\mathcal{L}_t = \{\mathbf{l}_1, \mathbf{l}_2, ..., \mathbf{l}_t\}$ represent the corresponding LiDAR observation. Therefore the joint posterior probability

density function by time step t is defined as

$$p\left(\mathcal{S}_t|\tilde{\mathbf{s}}_1,\mathcal{L}_t,\mathcal{M}_p\right)=p\left(\mathbf{s}_t|\mathcal{S}_{t-1},\mathbf{l}_t,\mathcal{M}_p\right)\cdot p\left(\mathcal{S}_{t-1}|\tilde{\mathbf{s}}_1,\mathcal{L}_{t-1},\mathcal{M}_p\right), \quad (3)$$

where $p\left(\mathcal{S}_{t-1}|\tilde{\mathbf{s}}_1,\mathcal{L}_{t-1},\mathcal{M}_p\right)$ denotes the probability distribution over pose sequence at timestamp $t-1$, while $p\left(\mathbf{s}_t|\mathcal{S}_{t-1},\mathbf{l}_t,\mathcal{M}_p\right)$ represents the probability distribution of the robot pose at timestamp t.

In the implementation, we make use of the constant velocity motion model $p\left(\mathbf{u}_{t-1}|\mathbf{s}_{t-1},\mathbf{s}_{t-2}\right)$ to provide the initial guess $\tilde{\mathbf{s}}_t$ of the robot pose $p\left(\tilde{\mathbf{s}}_t|\mathbf{s}_{t-1},\mathbf{u}_{t-1}\right)$, and $\mathbf{u}_{t-1}$ represents the motion change of the robot between timestamp $t-1$ and timestamp $t-2$. Therefore, the term $p\left(\mathbf{s}_t|\mathcal{S}_{t-1},\mathbf{l}_t,\mathcal{M}_p\right)(t>2)$ in equation (3) can be rewritten as

$$\begin{aligned}p\left(\mathbf{s}_t|\mathcal{S}_{t-1},\mathbf{l}_t,\mathcal{M}_p\right)&\triangleq p\left(\mathbf{s}_t|\mathbf{s}_{t-1},\mathbf{s}_{t-2},\mathbf{l}_t,\mathcal{M}_p\right)\\&=\int_{\tilde{\mathbf{s}}_t}p\left(\mathbf{s}_t|\tilde{\mathbf{s}}_t,\mathbf{l}_t,\mathcal{M}_p\right)\cdot p\left(\tilde{\mathbf{s}}_t|\mathbf{s}_{t-1},\mathbf{s}_{t-2}\right). \quad (4)\end{aligned}$$

B. 2D scan generation

Considering that the workspace of mobile robots mainly lies on the ground, accurate and real-time three degrees of freedom pose estimation is enough to complete various complex tasks. Therefore, we convert the dense 3D obstacle points into sparse but representative 2D obstacle points, which is enough for mobile robot pose tracking. This section describes the detailed process of 2D scan generation, as shown in Fig. 3.

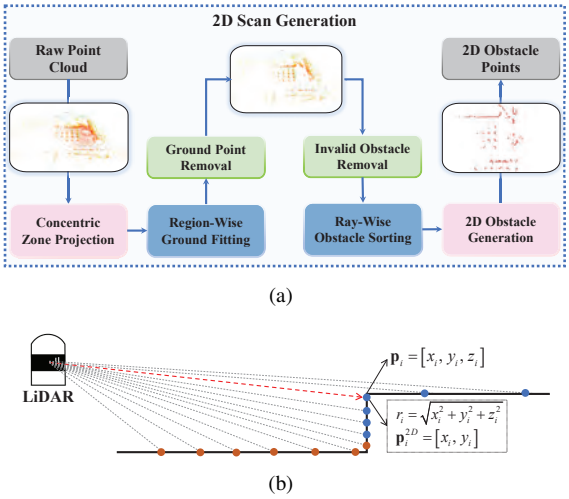


Fig. 3. (a) The process of 2D scan generation; (b) Ray-wise obstacle point selection. The orange point and the blue point represent the ground point and obstacle point, respectively. In addition, the obstacle point indicated by the red arrow is the final selection with the minimum range value.

At first, the point cloud is projected into a fan-shaped grid map using the concentric zone model. Ground plane fitting [44] is subsequently performed for each region to separate ground and obstacle points. Unlike previous work [45] that emphasized obstacle point recall, we prioritize accurate pose tracking using precise environmental information. Unreliable obstacle points in regions with poor ground fitting and high-altitude points are removed to enhance accuracy.

Afterward, the valid 3D obstacle points are transformed into 2D obstacle points through ray-wise sorting and 2D generation. Specifically, the number N of 2D obstacle points is determined by the LiDAR's horizontal angle resolution δ_a,

$$N=\lceil 2\pi/\delta_a\rceil. \quad (5)$$

Accordingly, 2D obstacle points are derived from ray-wise sorting, with N rays, each having multiple 3D points. For the i-th valid point $\mathbf{p}_i=[x_i,y_i,z_i]$, the range r_i is calculated as shown in Fig. 3(b), and the horizontal angle θ_i and ray index j are determined as follows

$$\theta_i=\mathbf{atan2}(y_i,x_i), j=\lfloor\theta_i/\delta_a\rfloor. \quad (6)$$

Based on the above, the ray $\mathbf{R}_j$ is denoted by $\{\mathbf{p}_1,\mathbf{p}_2,...,\mathbf{p}_k\}$ with k being the number of points. The final 2D point $\mathbf{p}_i^{2D}=[x_i,y_i]$ is based on the nearest 3D point ($\mathbf{p}i$ in Fig. 3(b)). $\mathbf{p}_i^{2D}$ is set as $[r_{max}\cdot\cos(j\delta_a),r_{max}\cdot\sin(j\delta_a)]$ with r_{max} being the given maximum observation distance.

C. Offline map construction

Based on the obtained 2D scan information, the process of offline map construction is described in detail in this section.

1) Grid map generation: Let $\mathbf{C}_i$ denote the 2D scan, represented by $\mathbf{C}_i=\{\mathbf{p}_1^{2D},\mathbf{p}_2^{2D},...,\mathbf{p}_N^{2D}\}$, corresponding to the sparse perception. While reliable static obstacles aid pose estimation, LiDAR noise and dynamic obstacles pose challenges. Additionally, efficiently fusing multiple observations, compactly representing the environment, and constructing the prior map are crucial tasks.

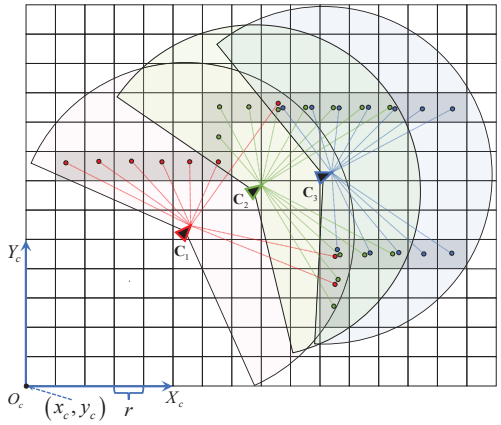
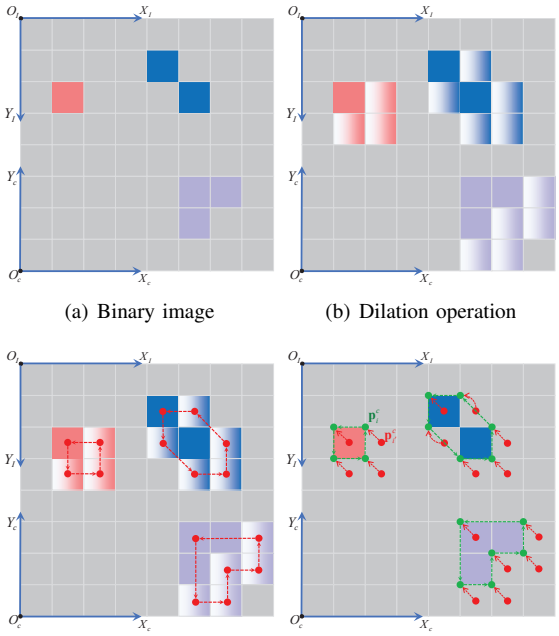


Fig. 4. The process of occupancy grid map generation. Continuous 2D scan observations $\mathbf{C}_1$, $\mathbf{C}_2$, and $\mathbf{C}_3$ are denoted by the red, green, and blue triangles, respectively, and the fan-shaped area represents the observation range. In addition, scan points obtained from continuous observations are represented by the color circles.

Accordingly, the occupancy grid map generation is introduced in Fig. 4, which is combined with the corresponding pose sequence $\mathcal{S}_t$. Therefore, equation (2) can be rewritten as

$$p\left(\mathcal{M}_p|\mathcal{C}_t,\mathcal{S}_t\right)=\int_{\mathcal{M}_g}p\left(\mathcal{M}_p|\mathcal{M}_g\right)\cdot p\left(\mathcal{M}_g|\mathcal{C}_t,\mathcal{S}_t\right), \quad (7)$$



(a) Binary image (b) Dilation operation

(c) Contour extraction (d) Contour translation

Fig. 5. The process of contour extraction for the grid map. The grids with color indicate the occupancy status, and red filled circles and green filled circles represent the contour vertices.

where $\mathcal{C}_t = \{\mathbf{C}_1, \mathbf{C}_2, ..., \mathbf{C}_t\}$ denotes the 2D scan sequence. The construction problem of grid map $\mathcal{M}_g$ is converted into the state estimation problem of each grid as follows

$$p\left(\mathcal{M}_g|\mathcal{C}_t, \mathcal{S}_t\right) = \prod_{i=1}^{N_g} p\left(\mathcal{C}_t, \mathcal{S}_t\right), \tag{8}$$

where g_i is the i-th grid, and N_g is the total number of grids. As shown in Fig. 4, the ray trace algorithm updates the grid map by connecting scan points to the sensor origin. Specifically, each grid's hits N_{hit} and passes N_{pass} are counted, and the occupancy probability of grid g_i is calculated as

$$p\left(g_i\right) = \frac{N_{hit}}{N_{hit} + N_{pass}}. \tag{9}$$

In the generation process, the size and the state of the grid map gradually update as new areas are observed. Finally, the occupancy probabilities of all grids are calculated, and each grid determines whether the current state is occupied based on the given threshold.

2) Polygon map construction: After the above generation process of the grid map, the binary image representing the static obstacles in the environment can be obtained. On this basis, the contour extraction process of the grid map is carried out to achieve a lightweight and compact representation.

In Fig. 5(a), the obtained grid map is transferred from the grid map coordinate system to the image coordinate system, denoted by X_c-O_c-Y_c and X_I-O_I-Y_I, respectively, and three different obstacles are taken to describe the contour extraction process in detail. The process of vectorized representation can be divided into three steps:

(1) As shown in Fig. 5(b), firstly, the original binary map is processed by dilation operation, and kernel size is 2, then the newly obtained binary map is used for contour extraction.

(2) Subsequently, the contour extraction process using the newly obtained binary map is performed, with the results depicted in Fig. 5(c) as red and directional contours.

(3) Moreover, the red contour in Fig. 5(c) does not match the actual obstacle information. The accurate final contours are obtained through contour translation, as shown in Fig. 5(d).

Furthermore, using points $\mathbf{p}_{i'}^c$ and $\mathbf{p}_i^c$ as an example, we explain how to obtain the contour vertex coordinates. Specifically, the image coordinates of $\mathbf{p}_{i'}^c$ are $(\text{row}_i, \text{col}_i)$, representing the row and column indexes. In the grid map coordinate system, $\mathbf{p}_{i'}^c = [x_{i'}^c, y_{i'}^c]$ is calculated as follows

$$x_{i'}^c = r \cdot (\text{row}_i + 0.5), \tag{10}$$

$$y_{i'}^c = r \cdot (N_{\text{col}} - \text{col}_i - 0.5), \tag{11}$$

where N_{col} denotes the number of columns. After the contour translation process, $\mathbf{p}_i^c = [x_i^c, y_i^c]$ is obtained as

$$x_i^c = x_{i'}^c - 0.5 \cdot r = r \cdot \text{row}_i, \tag{12}$$

$$y_i^c = y_{i'}^c + 0.5 \cdot r = r \cdot (N_{\text{col}} - \text{col}_i). \tag{13}$$

Taking into account the offset information of the grid map, the corresponding point coordinate $\mathbf{p}_i^w = [x_i^w, y_i^w]$ in the world coordinate system is calculated as follows

$$x_i^w = x_i^c + x_c, y_i^w = y_i^c + y_c. \tag{14}$$

Based on the above, a contour set $\mathcal{O} = \{\mathbf{O}_1, \mathbf{O}_2 ..., \mathbf{O}_{N_\mathcal{O}}\}$ is obtained for the environment, and $N_\mathcal{O}$ is the number of contours, while $\mathbf{O}_k$ is the k-th contour and represented by the vertex set $\{\mathbf{p}_1^w, \mathbf{p}_2^w, , ..., \mathbf{p}_{N_k}^w\}$ with N_k being the number of vertices. However, elongated and irregular contours can cause interference in subsequent data association and matching, further segmentation of the contours containing a large number of vertices into multiple small polygons is necessary. Accordingly, the proposed polygon segmentation process is shown in **Algorithm 1**, taking the original contour as input, the output is a number of polygons with fewer vertices than a given threshold. To be specific, the core process is shown in **lines 15-39**, and the function $\mathbf{JudgeInteraction}(\cdot)$ is used to judge whether the $\mathbf{Line}(\mathbf{p}_{I_1}, \mathbf{p}_{I_2})$ intersects with any edges on the contour. In addition, the contour is segmented into two polygons through the function $\mathbf{PolySegmentation}(\cdot)$, as shown in Fig. 6(b).

In particular, as depicted in Fig. 6(a), the example of polygon segmentation involves a contour with dozens of vertices arranged in counter-clockwise order. In the initial segmentation step, the vertices denoted as $\mathbf{p}_{10}^w$ and $\mathbf{p}_{34}^w$ serve as endpoints of the segmentation line, effectively dividing the contour into two roughly equal parts. This result is illustrated in Fig. 6(b). Ultimately, following two iterations of polygon segmentation, the original contour is divided into four distinct polygons, as illustrated in Fig. 6(d). After polygon segmentation and simplification, we make use of

the polygon map $\mathcal{M}_p = \{\mathbf{P}_1, \mathbf{P}_2 ..., \mathbf{P}_m\}$ to represent the environmental occupancy, and the i-th $\mathbf{P}_i$ is also denoted by the vertex set $\{\mathbf{p}_1^w, \mathbf{p}_2^w, , ..., \mathbf{p}_{N_i}^w\}$ with N_i being the number of vertices. Note that throughout the entire process of polygon segmentation and subsequent pose tracking, the counter-clockwise characteristic of the polygon vertices is preserved, as shown in Fig. 6.

3) Prior map representation: Given the polygon map $\mathcal{M}_p$, before commencing with pose tracking, the process of prior map representation is carried out, and this step is crucial for establishing a solid foundation for accurate tracking. Specifically, for the i-th polygon $\mathbf{P}_i = \{\mathbf{p}_1^w, \mathbf{p}_2^w, ..., \mathbf{p}_{N_i}^w\}$, the corresponding centroid point $\mathbf{p}_i^e = [x_i^e, y_i^e]$ is calculated as

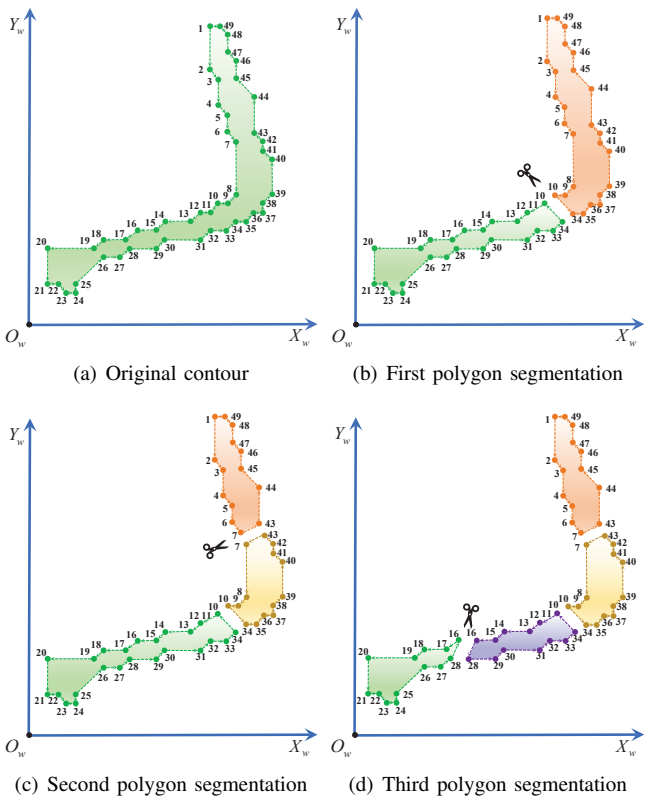
$$x_i^e = \frac{m_{10}}{m_{00}}, \; y_i^e = \frac{m_{01}}{m_{00}}, \tag{15}$$

where m_{10} and m_{01} represent the first-order moments about the x-coordinate and the y-coordinate, respectively, while m_{00} is the zeroth-order moment indicating the total area of $\mathbf{P}_i$. Therefore, we can calculate the corresponding centroid points for all polygons and form a centroid point cloud $\mathbb{C}_e = \{\mathbf{p}_1^e, \mathbf{p}_2^e, ..., \mathbf{p}_m^e\}$, accelerating the pose tracking process. On the other hand, all vertices contained in $\mathcal{M}_p$ form a vertex point cloud $\mathbb{C}_v = \{\mathbf{p}_1^v, \mathbf{p}_2^v, ..., \mathbf{p}_{N_v}^v\}$ with N_v being the number, and duplicate vertices are removed.

Algorithm 1 Polygon segmentation

Input: Original contour $\mathbf{O}_k = \{\mathbf{p}_1^w, \mathbf{p}_2^w, , ..., \mathbf{p}_{N_k}^w\}$, vertex number threshold N_v, and the segmentation distance threshold D_s.
Output: Polygon set $\mathcal{P} = \{\mathbf{P}_1, \mathbf{P}_2, ..., \mathbf{P}_l\}$ with l being the number of polygons.

The number of vertices N_k = size($\mathbf{O}_k$).
if $N_k < N_v$ **then**
  $\mathcal{P} \leftarrow \mathcal{P} \cup \{\mathbf{O}_k\}$
  Return
end if
Initialize the open_contour queue $\mathcal{O}_{open}$ with $\mathbf{O}_k$
while $\mathcal{O}_{open}$ is not empty **do**
  Obtain current open_contour $\mathbf{O}_c \leftarrow \mathcal{O}_{open}$.front(), $\mathcal{O}_{open}$.pop_front().
  N_c = size($\mathbf{O}_c$), and let δ denote the index interval, $\delta \leftarrow \lfloor N_c/2 \rfloor$.
  Let $\mathbf{O}_l$ and $\mathbf{O}_r$ represent two contours after segmentation.
  Let D_{min} and I_{min} denote the min distance and the corresponding index,
  and D_{min} is initialized with maximum value.
  $flag_S = False$, indicating whether segmentation has been performed.
  while $\delta \geqslant 2$ **do**
    for $i = 1 \rightarrow (N_c - \delta)$ **do**
      $I_1 \leftarrow i$, $I_2 \leftarrow (i + \delta)$
      Obtain two corresponding points $\mathbf{p}_{I_1}^w$ and $\mathbf{p}_{I_2}^w$
      $flag_I = \textbf{JudgeInteraction}(\mathbf{p}_{I_1}, \mathbf{p}_{I_2}, \mathbf{O}_c)$
      if $flag_I == True$ **then**
        Continue
      end if
      if $\textbf{DistancePoint}(\mathbf{p}_{I_1}^w, \mathbf{p}_{I_2}^w) < D_{min}$ **then**
        $D_{min} \leftarrow \textbf{DistancePoint}(\mathbf{p}_{I_1}^w, \mathbf{p}_{I_2}^w)$, $I_{min} \leftarrow i$
      end if
    end for
    if $D_{min} < D_s$ **then**
      $flag_S = True$
      $[\mathbf{O}_l, \mathbf{O}_r] \leftarrow \textbf{PolySegmentation}(\mathbf{p}_{I_{min}}^w, \mathbf{p}_{I_{min}+\delta}^w, \mathbf{O}_c)$
      if size($\mathbf{O}_l$) $< N_v$ **then**
        $\mathcal{P} \leftarrow \mathcal{P} \cup \{\mathbf{O}_l\}$
      else
        $\mathcal{O}_{open}$.emplace($\mathbf{O}_l$)
      end if
      if size($\mathbf{O}_r$) $< N_v$ **then**
        $\mathcal{P} \leftarrow \mathcal{P} \cup \{\mathbf{O}_r\}$
      else
        $\mathcal{O}_{open}$.emplace($\mathbf{O}_r$)
      end if
    end if
    $\delta \leftarrow \lfloor \delta/2 \rfloor$
  end while
  if $flag_S == False$ **then**
    $\mathcal{P} \leftarrow \mathcal{P} \cup \{\mathbf{O}_c\}$
  end if
end while



(a) Original contour (b) First polygon segmentation

(c) Second polygon segmentation (d) Third polygon segmentation

Fig. 6. The process of polygon segmentation. Specifically, the filled circles represent the vertices of the polygons, dashed lines represent the contour lines, color filled areas represent the polygon regions, and numbers refer to the index order. After three times of polygon segmentation (*i.e.*, a→b→c→d), the original contour is split into four non-intersecting polygons.

D. Online pose tracking

Once the prior map of the environment has been obtained, it can be utilized as prior information to facilitate effective and reliable pose tracking. Specifically, the newly proposed pose tracking approach encompasses the following detailed processes: feature extraction, point-polygon matching, and real-time pose tracking.

1) Feature extraction: Different from the previous offline map construction, the obtained 2D scan information is not immediately applied to pose tracking. Instead, it is processed to generate a more representative set of features with fewer elements. On the other hand, LiDAR odometry based on the LOAM (Lidar Odometry and Mapping) approach has achieved significant success in the field of mobile robotics and autonomous driving. Capitalizing on this established groundwork, we have adopted the LiDAR feature extraction process.

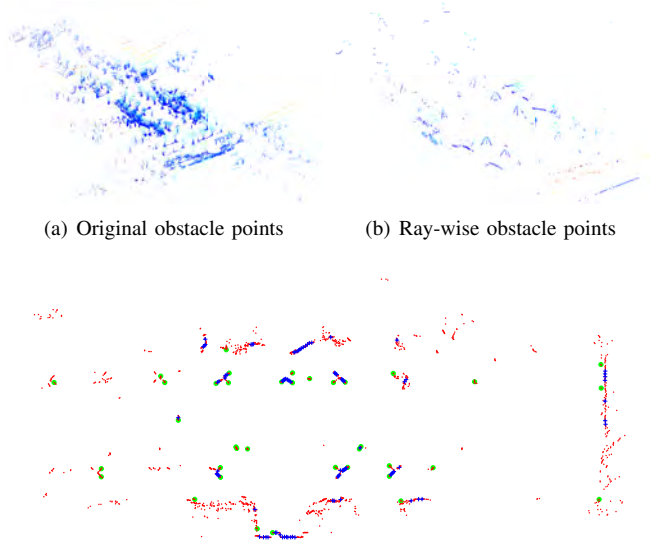
However, rather than extracting LiDAR features from dense point clouds, we opt to extract features from the

generated 2D scan data, following processes of ground removal and obstacle selection. Specifically, for the i-th 2D scan $\mathbf{C}_i = \{\mathbf{p}_1^{2D}, \mathbf{p}_2^{2D}, ..., \mathbf{p}_N^{2D}\}$, the smoothness value c_j of the j-th scan point is calculated with its neighbour scan points $\mathcal{S}$,

$$c_j = \left\| \sum_{k \in \mathcal{S}, k \neq j} \left(r_k^{2D} - r_j^{2D}\right) \right\|^2, \tag{16}$$

$$r_k^{2D} = \sqrt{(x_k)^2 + (y_k)^2}. \tag{17}$$

Afterward, according to the given smoothness threshold (σ_c and σ_e), the corner feature set $\mathbb{F}_c$ and edge feature set $\mathbb{F}_e$ can be obtained. Specifically, we exclusively employ the derived single-line scan for feature extraction, while corner features and edge features are identified as scan points exhibiting relatively lower and higher smoothness values, respectively, as shown in Fig. 7. Note that the obtained nearest obstacle points (Fig. 7(b)) retain their x-coordinates and y-coordinates, thereby compressing them into a 2D scan (Fig. 7(c)) for feature extraction, which is different from the conventional methodologies employed in 3D LiDAR odometry.



(a) Original obstacle points (b) Ray-wise obstacle points

(c) Representative feature points

Fig. 7. The process of sparse feature extraction. The original scan points are denoted by the red filled circles in (c), which are derived from the ray-wise obstacle points in (b), while the original obstacle points are illustrated in (a). In addition, the filled green circle and the blue cross represent the corner feature point and the edge feature point, respectively.

2) Point-polygon matching: Based on the prior map obtained through polygon map construction, the point-polygon matching process is executed in this subsection, leveraging two point clouds denoted as ($\mathbb{C}_e$ and $\mathbb{C}_v$), as well as two sets of feature points ($\mathbb{F}_c$ and $\mathbb{F}_e$). In addition, the polygon map $\mathcal{M}_p$ with counter-clockwise polygon vertices is also used for the point-to-edge matching step. Note that before implementing a matching process, it is necessary to ensure that the current number of observed corner features and edge features are greater than the thresholds τ_c and τ_e, respectively.

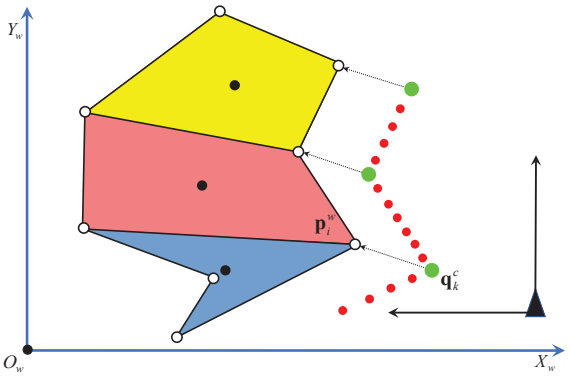


Fig. 8. The schematic illustration of data association for point-to-vertex. The polygon represents the occupancy information in $\mathcal{M}_p$, with its vertices indicated by white circles, and the black filled circle is the centroid point. In addition, the larger green circle represents the extracted corner feature point.

Specifically, the polygon map form utilized in this study is actually a lightweight and accurate description of environmental occupancy information. Consequently, during the processes of feature matching or pose estimation, it is essential that the feature points within the current 2D scan align with or closely conform to the polygons as much as possible.

Given the prior pose $\tilde{\mathbf{s}}_t = [\tilde{x}_t, \tilde{y}_t, \tilde{\theta}_t]$ regarding the current observation, it represents an imprecise estimation. Accordingly, we must formulate a cost function $\mathbf{F}(\tilde{\mathbf{s}}_t, \mathbb{F}_c, \mathbb{F}_e, \mathbb{C}_v, \mathbb{C}_e, \mathcal{M}_p)$ that incorporates the current features, pose hypotheses, and the prior map to accurately determine current pose

$$\mathbf{s}_t^* = \arg\min \mathbf{F}(\tilde{\mathbf{s}}_t, \mathbb{F}_c, \mathbb{F}_e, \mathbb{C}_v, \mathbb{C}_e, \mathcal{M}_p), \tag{18}$$

where $\mathbf{s}_t^*$ is the optimal estimation of the current pose, and the cost function is represented as follows

$$\begin{aligned} &\mathbf{F}(\tilde{\mathbf{s}}_t, \mathbb{F}_c, \mathbb{F}_e, \mathbb{C}_v, \mathbb{C}_c, \mathcal{M}_p) \\ &\triangleq \mathbf{F}_c(\tilde{\mathbf{s}}_t, \mathbb{F}_c, \mathbb{C}_v) + \mathbf{F}_e(\tilde{\mathbf{s}}_t, \mathbb{F}_e, \mathbb{C}_e, \mathcal{M}_p), \end{aligned} \tag{19}$$

where $\mathbf{F}_c(\cdot)$ and $\mathbf{F}_e(\cdot)$ represent the matching cost functions for point-to-vertex and point-to-edge, respectively. It is noteworthy that this study introduces an innovative approach, transforming the matching relationship between feature points and polygons into two distinct types of matching: point-to-vertex and point-to-edge, thereby enhancing the precision and adaptability of the feature matching technique.

Point-to-Vertex: In the realm of point-to-vertex matching, our methodology commences by constructing a KD-tree based on $\mathbb{C}_v$. Subsequently, assuming that the closest vertex retrieved from the KD-tree represents the accurate correspondence for each corner feature point, provided that the distance from the closest vertex is less than the given threshold ϵ_v.

For the retrieved closest vertex, all polygons containing these vertices are recalled. Afterward, the spatial relationship

between the current feature point and all polygons is analyzed. In instances where the point is found to be enclosed by a polygon, it will be excluded from the subsequent phase of cost function formulation. Otherwise, as shown in Fig. 8, the corner feature point $\mathbf{q}_k^c \in \mathbb{F}_c$ corresponds the vertex $\mathbf{p}_{k\to i}^w \in \mathbb{C}_v$, and these two points form a matching relationship,

$$\mathbf{p}_{k\to i}^w = \tilde{\boldsymbol{R}}_t \cdot \mathbf{q}_k^c + \tilde{\boldsymbol{t}}_t, \tag{20}$$

where

$$\tilde{\boldsymbol{R}}_t = \begin{bmatrix} \cos(\tilde{\theta}_t) & -\sin(\tilde{\theta}_t) \\ \sin(\tilde{\theta}_t) & \cos(\tilde{\theta}_t) \end{bmatrix}, \tilde{\boldsymbol{t}}_t = \begin{bmatrix} \tilde{x}_t \\ \tilde{y}_t \end{bmatrix}. \tag{21}$$

Accordingly, the mathematical expression of the Euclidean distance of matching pairs is as follows

$$\mathbf{F}_c(\tilde{\mathbf{s}}_t, \mathbb{F}_c, \mathbb{C}_v) = \sum_{k=1}^{N_c} \left\| \mathbf{p}_{k\to i}^w - \tilde{\boldsymbol{R}}_t \cdot \mathbf{q}_k^c - \tilde{\boldsymbol{t}}_t \right\|^2, \tag{22}$$

where N_c denotes the number of valid matching pairs formed by corner feature points.

Point-to-Edge: For the purpose of point-to-edge matching, a KD-tree is meticulously constructed from the point cloud denoted as $\mathbb{C}_e$, where each point within $\mathbb{C}_e$ corresponds to a specific polygon in $\mathcal{M}_p$ with counter-clockwise vertices. This KD-tree is subsequently employed to identify the K-nearest polygons associated with each edge feature point. In addition, the distance from the closest polygon should be less than the given threshold ϵ_p. In scenarios where a point is determined to be enclosed by the K-nearest polygons, as shown by $\mathbf{q}_{k_3}^e$ in Fig. 9, it is deliberately excluded from the subsequent phase of formulating the cost function.

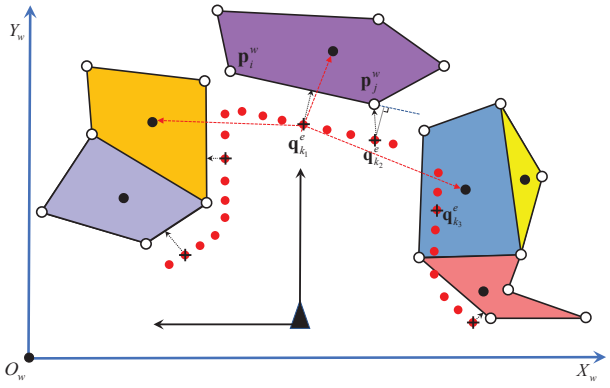


Fig. 9. The schematic illustration of data association for point-to-edge. The polygon represents the occupancy information in $\mathcal{M}_p$, with its vertices indicated by white circles, and the black filled circle is the centroid point. In addition, the black cross represents the extracted edge feature point, and three distinct forms of data association emerge from the edge feature points, marked by $\mathbf{q}_{k_1}^e$, $\mathbf{q}_{k_2}^e$, and $\mathbf{q}_{k_3}^e$, respectively.

On the other hand, two different constraint forms are constructed for edge feature points, as shown by $\mathbf{q}_{k_1}^e$ and $\mathbf{q}_{k_2}^e$ in Fig. 9. Firstly, for the edge feature point $\mathbf{q}_{k_2}$, find the polygon that is closest to it, as shown by the top polygon in Fig. 9. Subsequently, the next step involves identifying the edge on the polygon that is in closest proximity to the edge feature point. Finally, the identification of various constraint forms depends on whether the projection of the edge feature point onto the nearest edge falls precisely on the edge. Therefore, same as the form of point-to-vertex, the matching relationship between $\mathbf{q}_{k_2}^e$ and $\mathbf{p}_{k\to j}^w$ is established

$$\mathbf{F}_e^1 = \sum_{k_2=1}^{N_e^1} \left\| \mathbf{p}_{k_2\to j}^w - \tilde{\boldsymbol{R}}_t \cdot \mathbf{q}_{k_2}^e - \tilde{\boldsymbol{t}}_t \right\|^2, \tag{23}$$

where the number of valid matching pairs is denoted by N_e^1. In addition, the edge feature point $\mathbf{q}_{k_1}^e \in \mathbb{F}_e$ corresponds the edge $\{\mathbf{p}_{k_1\to i}^w, \mathbf{p}_{k_1\to j}^w\} \in \mathcal{M}_p$, and the matching relationship is established as follows

$$\mathbf{F}_e^2 = \sum_{k_1=1}^{N_e^2} \frac{\left\| (\mathbf{q}_{k_1}^w - \mathbf{p}_{k_1\to i}^w) \times \left(\mathbf{q}_{k_1}^w - \mathbf{p}_{k_1\to j}^w \right) \right\|^2}{\left\| \mathbf{p}_{k_1\to i}^w - \mathbf{p}_{k_1\to j}^w \right\|^2}, \tag{24}$$

where N_e^2 is number of valid matching pairs, and $\mathbf{q}_{k_1}^w$ is calculated as follows

$$\mathbf{q}_{k_1}^w = \tilde{\boldsymbol{R}}_t \cdot \mathbf{q}_{k_1}^e + \tilde{\boldsymbol{t}}_t. \tag{25}$$

In summary, the matching cost function of point-to-edge is represented as follows

$$\mathbf{F}_e(\tilde{\mathbf{s}}_t, \mathbb{F}_e, \mathbb{C}_e, \mathcal{M}_p) = \mathbf{F}_e^1 + \mathbf{F}_e^2. \tag{26}$$

On this basis above, in order to obtain the accurate pose estimation, ERPoT performs nonlinear optimization similar to iterative closest point (ICP) methods. This optimization terminates either upon detecting convergence or once the predefined maximum number of iterations has been reached.

3) Real-time pose tracking: In this subsection, the real-time pose tracking process is introduced, anchored by the matching cost function that correlates the current observation with the prior polygon map. Especially, during the initial phase of pose tracking, it is essential to provide an initial pose estimate for the current observation as prior information, which is denoted as $\tilde{\mathbf{s}}_1$. Subsequently, according to equation (18), the optimal estimation about $\mathbf{s}_1$ is derived.

Subsequently, for the real-time pose tracking, the continuous pose sequence is recursively obtained according to the continuous observation sequence and the prior polygon map. Taking the time step at $t(>2)$ as an example, the corresponding pose guess $\tilde{\mathbf{s}}_t$ is obtained according to $\mathbf{s}_{t-1}$ and $\mathbf{s}_{t-2}$ using a constant velocity motion model,

$$\tilde{\boldsymbol{R}}_t = \boldsymbol{R}_{t-1} \cdot \Delta\tilde{\boldsymbol{R}}_{t-1}, \tilde{\boldsymbol{t}}_t = \boldsymbol{R}_{t-1} \cdot \Delta\tilde{\boldsymbol{t}}_{t-1} + \boldsymbol{t}_{t-1}, \tag{27}$$

where $\Delta\tilde{\boldsymbol{R}}_{t-1}$ and $\Delta\tilde{\boldsymbol{t}}_{t-1}$ are calculated as

$$\Delta\tilde{\boldsymbol{R}}_{t-1} = \boldsymbol{R}_{t-2}^T \cdot \boldsymbol{R}_{t-1}, \tag{28}$$

$$\Delta\tilde{\boldsymbol{t}}_{t-1} = \boldsymbol{R}_{t-2}^T \cdot (\boldsymbol{t}_{t-1} - \boldsymbol{t}_{t-2}). \tag{29}$$

Therefore, the optimal pose estimation can be obtained according to equation (18) based on current pose guess $\tilde{\mathbf{s}}_t$, current observation, and the prior polygon map. Subsequently, the reliability of the current tracking result is assessed based on the discrepancy between the solved pose

information and the predicted pose information. At the same time, the distance and angle thresholds are set to λ_d and λ_a, respectively.

IV. Experiments

To show the superiority and effectiveness of the proposed ERPoT, both self-recorded datasets and publicly available datasets are used for the evaluation in this section. As shown in Fig. 10, the experimental environment exemplifies a typical campus setting, characterized by an abundance of architectural structures and lush vegetation. Moreover, publicly available datasets used in this evaluation include the KITTI dataset [46], the Newer College dataset [47], and the FusionPortable dataset [48], while corresponding environments and platforms are shown in Fig. 11.

Fig. 10. Experimental setup for the self-recorded dataset. In particular, the top figure depicts the experimental setting, which is representative of a typical campus environment. In addition, the bottom two figures show two dataset collection platforms, 204AKZ (Eagle) and FR-09 (YUHESEN), equipped with two different LiDAR sensors, named VLP-32C (Velodyne) and Pandar40P (HESAI), respectively.

TABLE I

Hardware and software specification

Robot	**204AKZ and FR-09**
Sensor	**VLP-32C and Pandar40P**
Processor	**Intel Core i7-11800H @2.3Ghz**
RAM	**32GB**
Operating system	**Ubuntu 20.04**
ROS	**Noetic**

Furthermore, the Robot Operating System (ROS) is utilized for both hardware and software implementations. The code for constructing the polygon map and performing pose tracking is written in C/C++. Experiments are conducted on a computer with specifications detailed in Tab. I and the system parameters of ERPoT are shown in Tab. II. The approach has been benchmarked against six other methods. It is important to note that during the pose tracking process, the estimation of the current pose relies solely on the prior map (including the point cloud map, mesh map, distance field map, and the proposed polygon map) and data obtained from the LiDAR sensor.

TABLE II

Systems parameters

Parameter:	**value**
Grid map resolution (r)	
- *Small scenes*	0.05m
- *Large scenes*	0.10m
Pose tracking	
- *Vertex data association threshold* (ϵ_v)	2.0m
- *Polygon data association threshold* (ϵ_p)	3.0m
- *Distance difference threshold* (λ_d)	1.0m
- *Angle difference threshold* (λ_a)	0.2rad
Feature extraction	
- *Corner smoothness threshold* (σ_c)	10.0
- *Edge smoothness threshold* (σ_e)	0.1
- *Corner feature number threshold* (τ_c)	10
- *Edge feature number threshold* (τ_e)	50

`ALOAM_MCL`: A-LOAM[1] is an advanced implementation of LOAM [14], which makes use of Eigen and Ceres Solver to simplify code structure, providing the real-time odometry information in pure LiDAR mode. In addition, combined with the point cloud-based Monte Carlo localization[2] (MCL), we can obtain a probabilistic 6-DOF localization system ALOAM_MCL for mobile robots with only LiDAR sensor.

`FLOAM_MCL`: F-LOAM[3] [28] represents a streamlined and cost-efficient evolution of A-LOAM and LOAM, offering improved computational efficiency while maintaining robust LiDAR odometry for mobile robots. In addition, based on the point cloud-based MCL and the prior map, the process of global pose tracking FLOAM_MCL can be carried out.

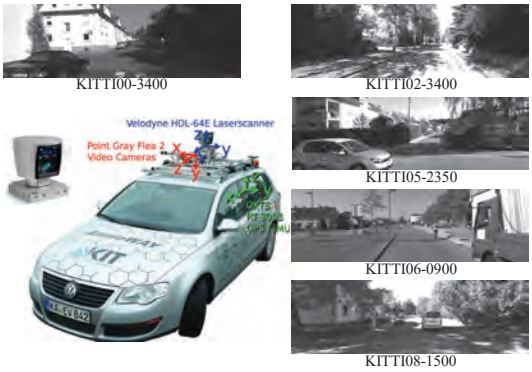
`KISS_MCL`: KISS-ICP[4] [13] is a LiDAR odometry pipeline that just works on most of the cases without tuning any parameter. This system stands out as a remarkably effective solution, noted for its simplicity in implementation and versatility in operation across diverse environmental settings with different LiDAR sensors. Afterward, the global pose tracking approach called KISS_MCL is obtained.

`HDL_localization`: HDL_localization[5] [39] is a ROS package for real-time 3D localization with different LiDAR sensors, such as Velodyne HDL-32E and VLP-16. Based on the voxelized GICP (VGICP) algorithm[6] [11], this package performs Unscented Kalman Filter-based pose estimation.

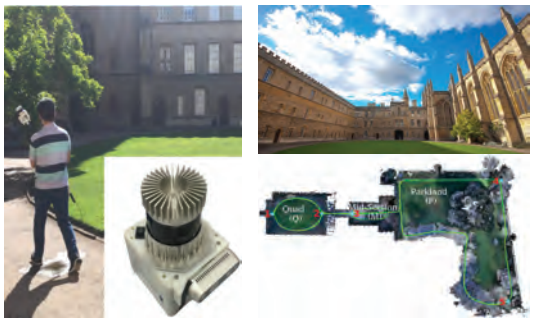
`SM_MCL`: The work in [20] provides a ROS package[7] of 3D LiDAR-based pose tracking for mobile robots. It acknowledges the complementary strengths and weaknesses of Monte Carlo Localization (MCL) and Scan Matching (SM), integrating these two sophisticated localization methods. The fusion of MCL and SM within SM_MCL is innovative, allowing the package to operate effectively under pure LiDAR mode.

`Range_MCL`: The range image provides a lightweight 3D

[1]https://github.com/HKUST-Aerial-Robotics/A-LOAM
[2]https://github.com/at-wat/mcl_3dl
[3]https://github.com/wh200720041/floam
[4]https://github.com/PRBonn/kiss-icp
[5]https://github.com/koide3/hdl_localization
[6]https://github.com/koide3/fast_gicp
[7]https://github.com/NaokiAkai/mcl3d_ros



(a) KITTI dataset



(b) Newer College dataset

(c) FusionPortable dataset

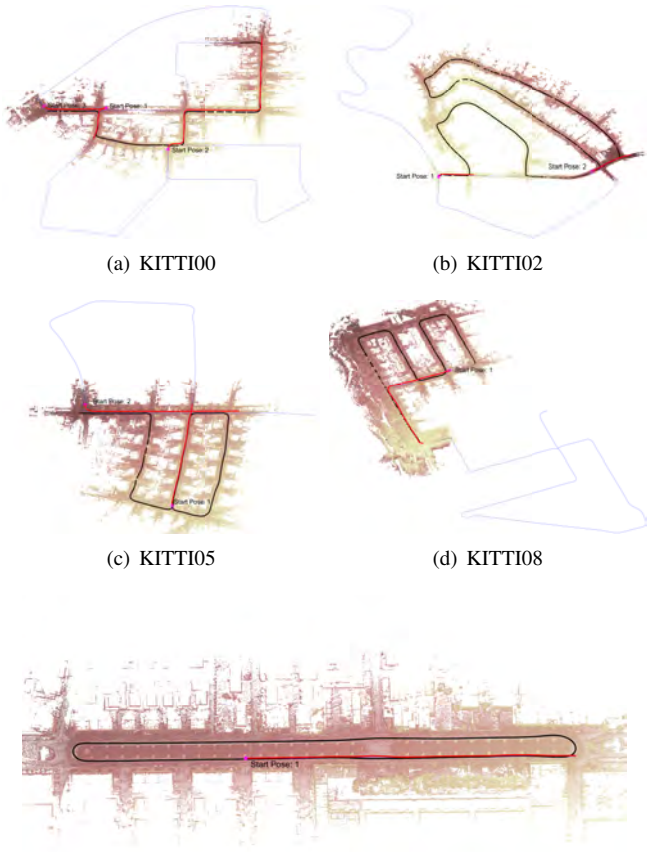
Fig. 11. The experimental environments and platforms for dataset collection of two publicly available datasets. For the KITTI dataset, the LiDAR sensor HDL-64E (Velodyne) is used to capture the corresponding point cloud dataset with typical street and road environments. The Newer College dataset is acquired using the handheld LiDAR sensor os1-64 (Ouster) at typical walking speeds through New College, Oxford for nearly 6.7 km. Additionally, the FusionPortable dataset, captured using Ouster os1-128, is a typical campus-scene dataset that includes both indoor and outdoor environments.

LiDAR scan representation, and the mesh map is more compact than the point cloud map. In [19], a novel sensor model combined with MCL is used for 3D LiDAR-based global localization and pose tracking [8]. This model compares current LiDAR range images to synthetic ones from the mesh to adjust particle weights, offering a simple yet versatile approach.

[8]https://github.com/PRBonn/range-mcl

A. Experimental configurations

In the pursuit of pose tracking evaluation, we meticulously select segments from each dataset to construct prior maps, which act as essential baselines for the localization process. Specifically, we make use of the provided segment sequence including ground truth values and corresponding point clouds to transform all point clouds into the world coordinate system, thereby creating a prior point cloud map. Additionally, to manage the large scale of the point cloud map, we apply voxel filtering to facilitate efficient online pose tracking, the final prior map is used for ALOAM_MCL, FLOAM_MCL, KISS_MCL, and HDL_localization. Based on the obtained point cloud map, the distance field representation for SM_MCL is generated using the provided open-source tool. Similarly, the prior mesh map for Range_MCL is acquired using the open-source code.



(a) KITTI00 (b) KITTI02

(c) KITTI05 (d) KITTI08

(e) KITTI06

Fig. 12. The experimental configurations of the KITTI dataset. The blue thin line represents the complete trajectory of the dataset, and the black thick line represents the trajectory used to construct the prior map. In addition, the trajectory denoted by the red thick line is used for evaluation of pose tracking, and the corresponding start pose is marked by the magenta pentagram.

For the KITTI dataset, sequences 00, 02, 05, 06, and 08 have been deliberately selected for our evaluative purposes, leveraging their overlapping trajectories to ensure a reliable and comprehensive assessment of pose tracking

performance. On the other hand, it is crucial to recognize that although the KITTI dataset remains the famous vision benchmark, the inconsistencies in the baselines are on the order of meters since the ground truth system relies only on RTK-GPS measurements [26]. Therefore, we combine point cloud registration with the ground truth provided by the KITTI dataset to further optimize the ground truth of the dataset for subsequent performance evaluation. The detailed experimental configurations are shown in Fig. 12, and the corresponding LiDAR sequences are shown in Tab. III.

TABLE III
EXPERIMENTAL CONFIGURATIONS OF KITTI DATASET

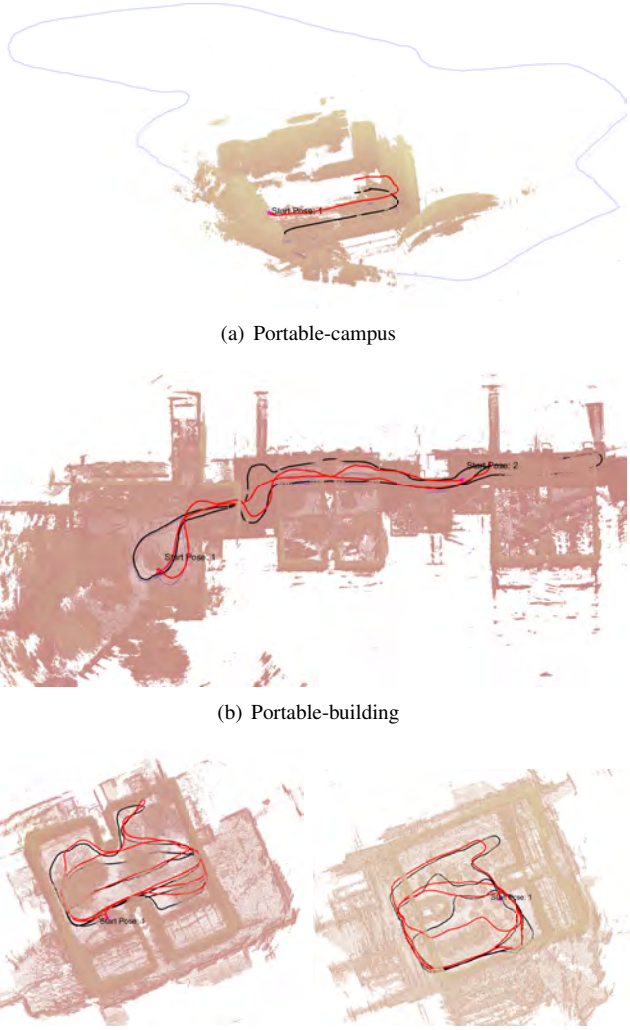
Dataset No.	Map construction	Pose tracking
00	0000-1000	1540-1650, 3360-3860, 4420-4540
02	0920-3408	4180-4280, 4549-4649
05	0000-1300	1301-1581, 2300-2650
06	0000-0833	0834-1100
08	0000-1306	1400-1850

Fig. 13. The experimental configurations of the Newer College dataset.

For the Newer College dataset, a comprehensive collection of 26559 LiDAR frames is meticulously captured throughout the environment, as depicted in Fig. 11(b). The sequence of LiDAR frames within the range of $[8700, 14550]$ is utilized to construct the prior map essential for pose tracking, represented by the black thick trajectory in Fig. 13. Furthermore, the sequences $[500, 8699]$ and $[14551, 25901]$ are used for experimental evaluation of pose tracking, denoted by the red thick trajectories in Fig. 13.

For the FusionPortable dataset, the experimental configurations are illustrated in Fig. 14. Specifically, for the Portable-canteen and Portable-garden datasets, which encompass both day and night scenes, the night datasets (20220215_canteen_night and 20220215_garden_night) are used to construct the prior maps, while the day datasets (20220216_canteen_day and 20220216_garden_day) are employed for pose tracking evaluation. For Portable-campus (20220226_campus_road_day), the sequence of LiDAR frames within the range $[1, 1100]$ is used for prior map construction, and the sequence $[11063, 12263]$ is utilized for experimental evaluation. For Portable-building, which includes both indoor and outdoor scenes, the range $[1201, 4500]$ is used for prior map construction, and the sequences $[1, 1200]$ and $[4501, 5996]$ are used for evaluation.

To further validate the effectiveness of the proposed approach, we have utilized two different platforms with different LiDAR sensors to collect multiple datasets at various times in the environment shown in Fig. 10. Specifically, for Self data-01, two sequences denoted by 'Start Pose: 1' and 'Start Pose: 2' in Fig. 15(a) are recorded at the same time (2021-09-07) as the sequence used to construct the prior map. While the third sequence denoted by 'Start Pose: 3' is recorded a year and a half later (2023-03-05) to verify the long-term viability of the proposed polygon map and ERPoT. On the other hand, considering the complexity of the actual environment, especially scenarios involving uphill and downhill sections, and Self dataset-02 has been recorded via the golf cart equipped with VLP-32, as shown in Fig. 10.



(a) Portable-campus

(b) Portable-building

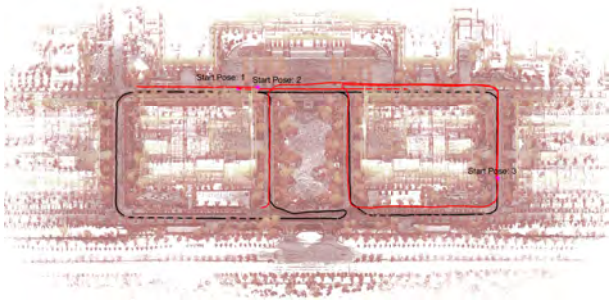
(c) Portable-garden

(d) Portable-canteen

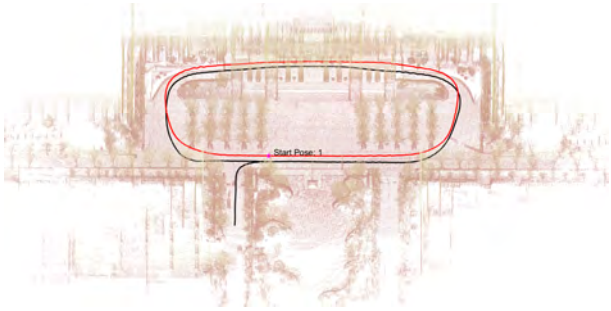
Fig. 14. The experimental configurations of the FusionPortable dataset.

B. Qualitative experimental results

In this section, the qualitative experimental results of pose tracking based on the prior polygon map are provided. At first, based on the experimental configurations for self-recorded datasets and publicly available datasets, the corresponding prior polygon maps for the proposed ERPoT are obtained. Subsequently, the pose tracking process based on



(a) Self dataset-01



(b) Self dataset-02

Fig. 15. The experimental configurations of self-recorded dataset.

these prior maps is executed and illustrated in Fig. 16.

For Self dataset-01, the process of pose tracking is represented by the three left subfigures in Fig. 16(a), and the obtained trajectory is smooth. As shown in the right subfigure in Fig. 16(a), the obtained 2D scan information is enough to describe the surrounding environmental information, including vegetation and buildings, which has a good matching relationship with the prior polygon map. Furthermore, within the Self dataset-02 environment, which includes uphill and downhill terrains, the proposed approach, while estimating only the 3-DoF (x, y, θ) of the pose, can still accurately estimate the pose during uphill and downhill movements, as demonstrated by the smooth trajectory in the Fig. 16(b).

On the other hand, for the KITTI00 dataset, our proposed ERPoT can realize effective and reliable pose tracking, even in scenarios where the collection platform navigates at high velocities within road environments. This capability is evident from the smooth trajectory illustrated in Fig. 16(c). In addition, in both Fig. 16(d) and Fig. 16(e), for the handheld data collection platform with different LiDAR sensors, the smooth pose estimation result can also be obtained.

Drawing from the aforementioned qualitative experimental outcomes, the proposed approach ERPoT has demonstrated its capacity to achieve effective and reliable pose tracking.

C. Experimental evaluation metrics

Before the detailed evaluation, this section provides an overview of the qualitative measures used to assess performance. A comprehensive set of metrics evaluates the pose tracking process based on a lightweight polygon map.

Translation error and rotation error: For pose tracking using a prior map, the system has a global reference with a global position and yaw angle. The accuracy of pose estimation is evaluated using the Absolute Pose Error (APE) [49]. In this study, the i-th estimated pose is denoted by $\mathbf{s}_i = [x_i, y_i, \theta_i]$, and the ground truth is $\hat{\mathbf{s}}_i = \left[\hat{x}_i, \hat{y}_i, \hat{\theta}_i\right]$, then the error between $\hat{\mathbf{s}}_i$ and $\mathbf{s}_i$ can be parameterized as $\Delta\mathbf{s}_i = [\Delta x_i, \Delta y_i, \Delta\theta_i]$, and satisfies the following equation

$$\boldsymbol{T}\left(\Delta\mathbf{s}_i\right) = \boldsymbol{T}(\hat{\mathbf{s}}_i)^{-1} \cdot \boldsymbol{T}\left(\mathbf{s}_i\right), \tag{30}$$

where the transformation $\boldsymbol{T}(\cdot) \in SE(2)$.

Runtime of pose tracking: In the realm of approach performance evaluation, runtime is a crucial metric, reflecting the efficiency of algorithms in processing information in various environments. Assessing runtime is essential for understanding the practical applicability of an approach, especially in time-sensitive environments. In comparative experiments, we report only CPU times, focusing on CPU performance to ensure consistency across different algorithms. The times cover the full pose tracking cycle, including data preprocessing, feature extraction, feature matching, and pose estimation.

Size of the prior map: More importantly, the prior map, a foundational element for accurate pose tracking, is designed to balance detail and efficiency, especially in large-scale, resource-constrained environments. We advocate for a map that is rich in essential features yet optimized for minimal storage and swift retrieval, facilitating real-time pose estimation without compromising computational resources.

D. Quantitative comparative results using public datasets

Based on the experimental configurations and metrics, comparative experiments are conducted between ERPoT and six other approaches. The prior polygon and point cloud maps are constructed using the LiDAR sequences listed in Tab. III for pose tracking. For SM_MCL, a distance field representation, which requires significant precomputation based on the point cloud map, is used for efficient pose tracking. In contrast, the polygon maps for KITTI sequences 00, 02, 05, 06, and 08, shown in Figure 17, are lightweight and compact. They capture essential spatial cues while omitting unnecessary details. This design allows ERPoT to operate with minimal memory usage, which is crucial for applications with hardware limitations.

Furthermore, pose tracking is conducted using the prior map, with the effect for KITTI00 shown in Fig. 16(c). More importantly, Tab. IV shows the quantitative comparative results on KITTI dataset, with optimal results in bold and failed cases indicated by '$-$'. During the pose tracking experiments for each test sequence, success is confirmed when the final pose error remains within the specified limits: translational error within $200\mathrm{cm}$ and rotational error within $20°$. Additionally, the pose error during the process must be less than $500\mathrm{cm}$ and $30°$. This criterion ensures that only the sequences meeting these standards are subjected to further evaluation.

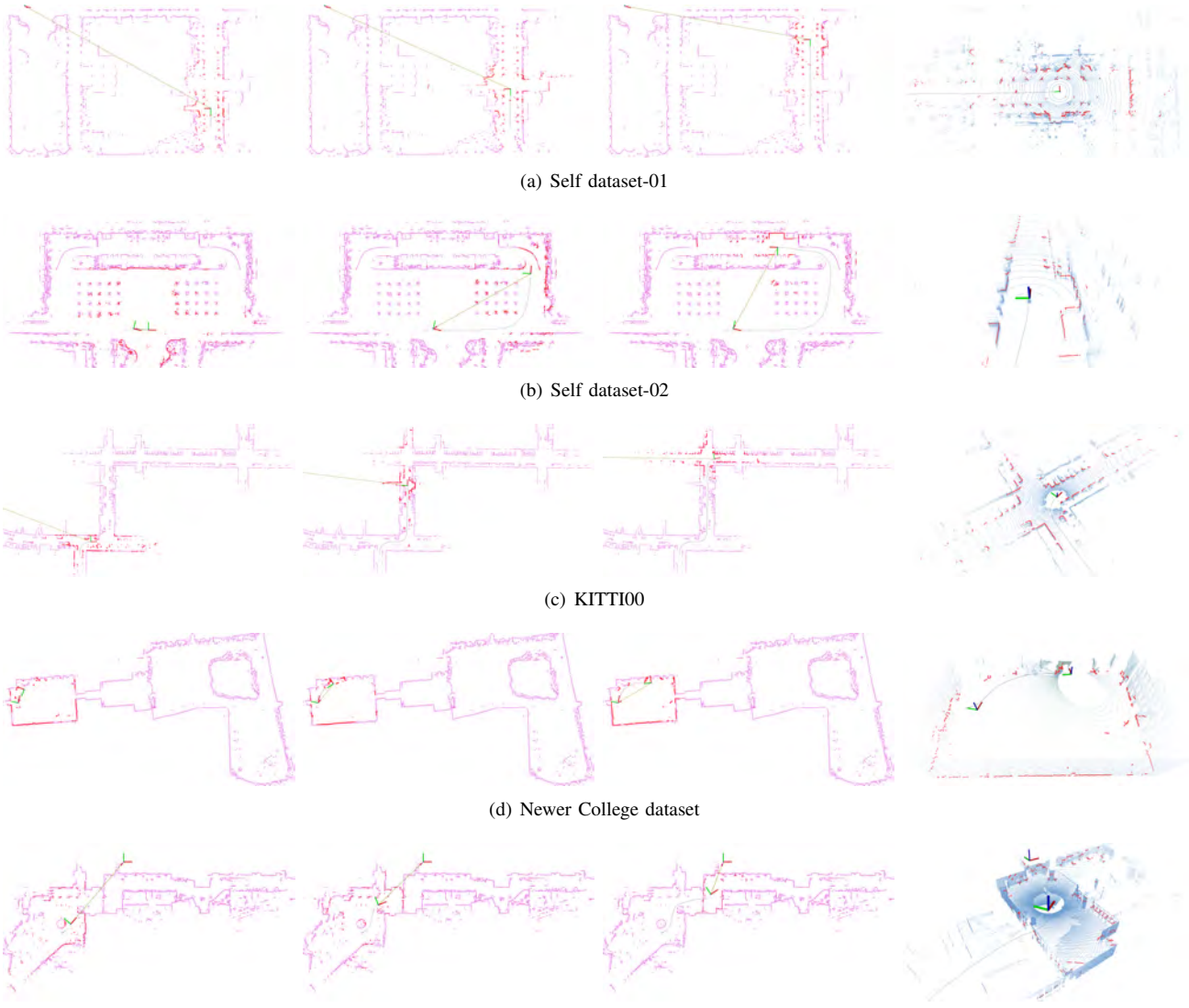
As shown in Tab. IV, ERPoT successfully achieved pose tracking on nine test sequence, which uses a novel polygon-

based prior map for compact and lightweight storage. Across all five test environments, the map size is consistently under 1 MB, demonstrating the efficiency and practicality of our method, and the number of polygons/vertices is displayed after the map size. Furthermore, regarding translational and rotational errors, the proposed approach has also demonstrated optimal performance. The mean values for translational and rotational errors are both below 20cm and $0.5°$, respectively, across all test scenarios, underscoring the accuracy and precision of our method. In addition, ERPoT demonstrates exceptional runtime performance, reliably achieving a consistent processing time of approximately 40ms with high stability.

On the other hand, comparative experiments on the Newer College dataset are conducted, with the polygon map shown in Fig. 18. The campus buildings are represented by compact polygons to create a lightweight prior map, as detailed in the right subfigure. Two test sequences (Newer College-1 and Newer College-2) are evaluated for pose tracking, with quantitative results presented in Tab. V.

The Newer College dataset, collected via a handheld platform, contains noisy LiDAR data due to movement. Combined with extensive vegetation, these factors create significant challenges for pure LiDAR pose tracking. As a result, methods like FLOAM_MCL, HDL_localization, and SM_MCL fail to achieve effective tracking in both test sequences. However, our proposed ERPoT maintains successful pose tracking capabilities, even though it exhibits slightly higher translational and rotational errors. Moreover, the proposed ERPoT also demonstrates the superior performance in terms of runtime.

To further validate the effectiveness and reliability of



(a) Self dataset-01

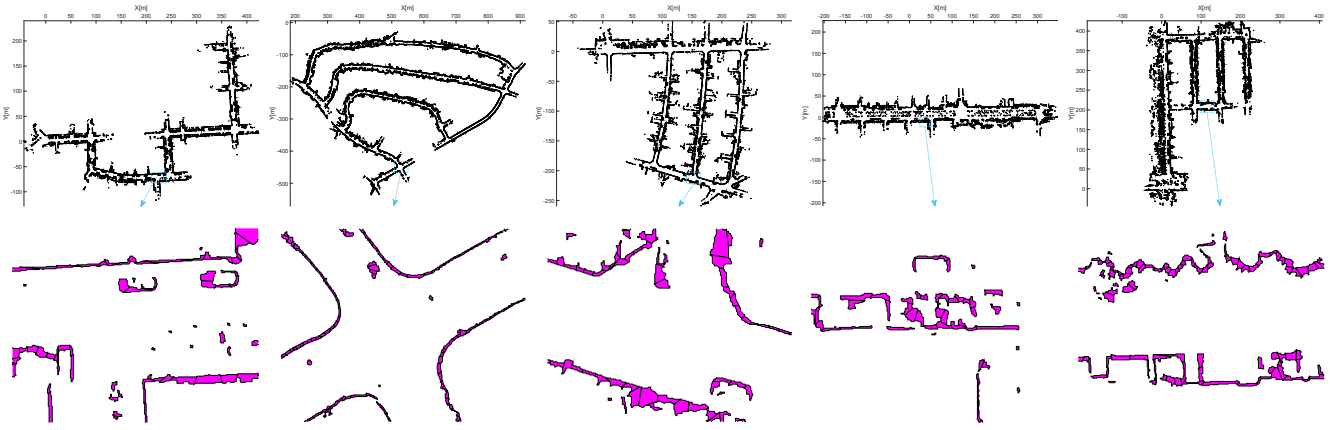
(b) Self dataset-02

(c) KITTI00

(d) Newer College dataset

(e) Portable-building dataset

Fig. 16. The experimental results of the pose tracking process based on the prior maps for both self-recorded datasets and publicly available datasets. The thin black line denotes the trajectory of pose tracking, and the 2D scan information based on the nearest obstacle selection is represented by the red point, while all the magenta polygons make up the prior map. Furthermore, the three left subfigures in each row depict the process of pose tracking, and the right subfigure describes the 2D scan information corresponding to the third subfigure.



(a) KITTI00 (b) KITTI02 (c) KITTI05 (d) KITTI06 (e) KITTI08

Fig. 17. Polygon maps for the KITTI dataset. Each polygon map is composed of multiple polygons and the corresponding centroid points, marked by the magenta filled polygons and black points. In addition, the area within the blue box is enlarged for display, as shown in the bottom subfigure. Due to the utilization of vector graphics for map rendering, the polygon map details can be viewed upon zooming in.

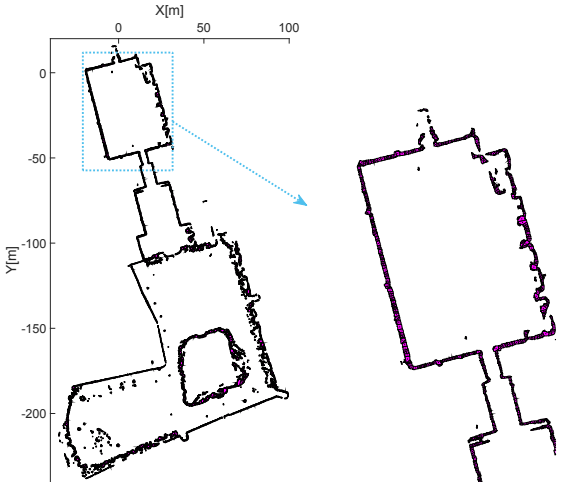
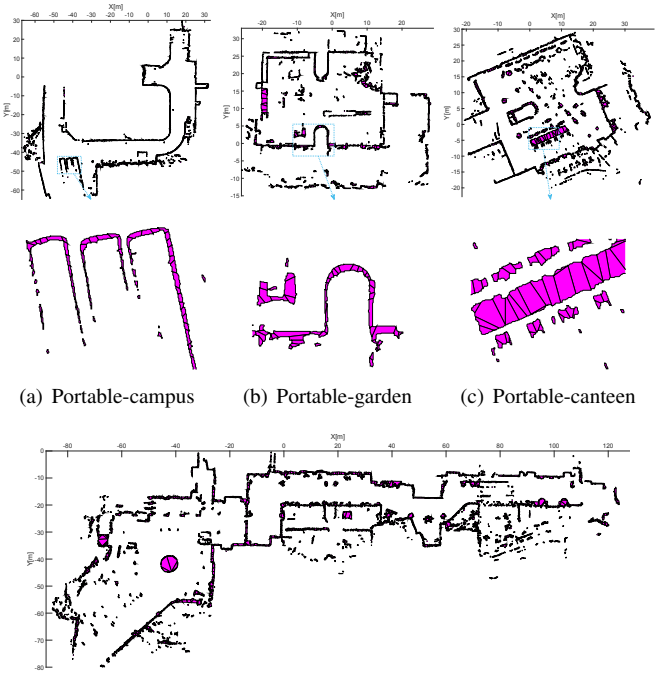


Fig. 18. Polygon map for the Newer College dataset.



(a) Portable-campus (b) Portable-garden (c) Portable-canteen

(d) Portable-building

Fig. 19. Polygon maps for the FusionPortable dataset.

ERPoT, additional comparative experiments are conducted using the Portable-campus and Portable-building datasets. The polygon maps for these scenarios are shown in Fig. 19(a) and Fig. 19(d), with results detailed in Tab. VI. Notably, the prior map size ranges from 0.1 to 0.2 MB, demonstrating remarkable performance. ERPoT also shows superior performance in translational error, rotational error, and runtime.

E. Comparative experiments for indoor environments

To further validate the effectiveness of ERPoT in indoor environments, comparative experiments are conducted using the Portable-garden and Portable-canteen datasets. These two scenarios represent typical indoor garden and canteen environments, respectively, and include dynamic pedestrian information. Detailed information is presented in Fig. 11(c).

The prior polygon maps are generated from the nighttime LiDAR sequences, as shown in Fig. 19(b) and Fig. 19(c). Notably, the prior map size of ERPoT remains exceptionally compact, even under 0.1MB in the garden environment. Comparative results are presented in Tab. VII, highlighting the challenges of pose tracking in complex indoor settings, which lead to the failure of some methods like ALOAM_MCL, HDL_localization, and SM_MCL. Despite these challenges, ERPoT demonstrates outstanding performance, particularly in runtime, achieving stable results with a 128-line LiDAR.

F. Quantitative comparative results using self datasets

Furthermore, to verify the effectiveness of the proposed approach in other complex operating environments, two self-recorded datasets (Self dataset-01 and Self dataset-02) are used for comparative evaluation of pose tracking. For Self

TABLE IV

EXPERIMENTAL RESULTS OF DIFFERENT APPROACHES USING THE KITTI DATASET

Dataset No.	Approach name	Prior map size (MB) ↓	Translational error (cm)			Rotational error (deg)			Runtime (ms)		
			Max ↓	Mean ↓	RMSE ↓	Max ↓	Mean ↓	RMSE ↓	Max ↓	Mean ↓	SD↓
KITTI00-1	ALOAM_MCL	56.44	92.85	58.44	63.70	2.72	0.96	1.16	221.08	175.54	26.91
	FLOAM_MCL		136.08	61.08	64.49	5.24	1.35	1.78	156.81	69.51	14.74
	KISS_MCL		74.88	49.18	51.27	4.66	1.76	2.11	**58.58**⋆	43.09	7.19
	HDL_localization		361.09	24.95	60.52	8.36	0.62	1.48	559.63	68.94	81.78
	SM_MCL	6709.49	99.72	33.00	38.15	2.07	0.55	0.66	68.03	**36.09**⋆	5.60
	Range_MCL	12.67	100.11	73.31	74.76	6.07	2.40	2.83	135.73	132.13	**4.59**⋆
	ERPoT(ours)	**0.28**⋆(2584/34708)	**47.96**⋆	**13.14**⋆	**16.33**⋆	**0.79**⋆	**0.38**⋆	**0.43**⋆	100.65	47.44	8.61
KITTI00-2	ALOAM_MCL	56.44	158.69	40.92	47.96	5.76	1.24	1.55	261.92	179.10	56.19
	FLOAM_MCL		138.17	49.73	58.74	4.34	1.19	1.47	170.37	76.94	12.61
	KISS_MCL		208.34	62.20	81.73	5.35	1.22	1.53	84.26	42.58	8.92
	HDL_localization		149.60	**7.72**⋆	11.32	5.08	0.49	0.67	334.79	61.64	46.85
	SM_MCL	6709.49	86.20	21.69	25.17	1.17	0.32	0.38	**50.33**⋆	**36.76**⋆	**4.75**⋆
	Range_MCL	12.67	–	–	–	–	–	–	–	–	–
	ERPoT(ours)	**0.28**⋆(2584/34708)	**23.66**⋆	8.36	**9.53**⋆	**0.78**⋆	**0.21**⋆	**0.26**⋆	61.55	40.81	5.67
KITTI00-3	ALOAM_MCL	56.44	186.50	114.40	125.01	4.78	1.38	1.82	225.06	175.33	46.56
	FLOAM_MCL		170.32	113.97	123.72	5.32	1.44	1.84	199.90	80.45	18.15
	KISS_MCL		265.02	131.44	152.89	8.28	1.82	2.47	65.65	44.06	9.13
	HDL_localization		48.52	**8.83**⋆	**10.79**⋆	3.72	0.31	0.74	444.92	77.79	75.35
	SM_MCL	6709.49	51.84	19.46	22.20	0.79	0.29	0.33	**59.71**⋆	**38.39**⋆	5.68
	Range_MCL	12.67	152.38	112.33	118.33	3.66	1.09	1.39	135.92	132.38	**1.72**⋆
	ERPoT(ours)	**0.28**⋆(2584/34708)	**34.46**⋆	12.88	16.64	**0.41**⋆	**0.12**⋆	**0.15**⋆	61.71	45.28	6.64
KITTI02-1	ALOAM_MCL	110.47	130.04	74.96	76.71	5.28	1.33	1.63	246.59	180.41	44.14
	FLOAM_MCL		130.24	69.36	73.32	4.68	1.19	1.55	202.30	81.64	19.11
	KISS_MCL		170.96	84.73	90.04	5.02	1.59	1.95	**60.13**⋆	**43.01**⋆	7.13
	HDL_localization		–	–	–	–	–	–	–	–	–
	SM_MCL	13838.15	75.97	19.10	22.83	0.69	0.31	0.35	69.77	36.27	9.26
	Range_MCL	27.38	–	–	–	–	–	–	–	–	–
	ERPoT(ours)	**0.85**⋆(7582/106065)	**33.13**⋆	**11.58**⋆	**13.39**⋆	**0.64**⋆	**0.23**⋆	**0.26**⋆	68.50	43.83	**6.84**⋆
KITTI02-2	ALOAM_MCL	110.47	–	–	–	–	–	–	–	–	–
	FLOAM_MCL		–	–	–	–	–	–	–	–	–
	KISS_MCL		181.80	140.22	142.50	5.36	1.97	2.34	76.80	**42.29**⋆	11.26
	HDL_localization		–	–	–	–	–	–	–	–	–
	SM_MCL	13838.15	–	–	–	–	–	–	–	–	–
	Range_MCL	27.38	–	–	–	–	–	–	–	–	–
	ERPoT(ours)	**0.85**⋆(7582/106065)	**83.44**⋆	**18.70**⋆	**22.10**⋆	**0.85**⋆	**0.29**⋆	**0.35**⋆	**74.72**⋆	48.40	**8.01**⋆
KITTI05-1	ALOAM_MCL	70.93	63.42	32.76	35.56	3.60	1.15	1.41	254.67	177.73	52.50
	FLOAM_MCL		127.64	40.94	50.25	6.08	1.25	1.58	189.57	77.71	14.50
	KISS_MCL		96.40	48.07	53.91	3.96	1.16	1.43	80.88	48.65	9.70
	HDL_localization		149.28	8.12	19.02	4.02	0.32	0.57	429.17	56.83	58.34
	SM_MCL	6647.48	44.22	15.55	17.78	**1.52**	0.35	0.41	**55.58**⋆	**34.17**⋆	5.05
	Range_MCL	23.37	–	–	–	–	–	–	–	–	–
	ERPoT(ours)	**0.36**⋆(3321/44174)	**25.46**⋆	**7.94**⋆	**9.01**⋆	1.58	**0.24**⋆	**0.33**⋆	58.91	42.63	**4.71**⋆
KITTI05-2	ALOAM_MCL	70.93	101.97	43.56	50.42	5.46	1.28	1.67	269.04	179.49	60.88
	FLOAM_MCL		216.58	65.23	85.12	7.99	1.22	1.66	215.86	83.66	14.59
	KISS_MCL		117.28	51.24	60.81	4.52	1.21	1.57	80.50	50.80	9.57
	HDL_localization		103.83	10.68	13.14	1.86	0.33	0.45	512.56	56.26	46.64
	SM_MCL	6647.48	40.92	15.30	17.87	0.95	0.43	0.48	81.48	43.76	8.43
	Range_MCL	23.37	176.95	83.49	91.05	5.80	1.32	1.66	235.29	151.41	88.27
	ERPoT(ours)	**0.36**⋆(3321/44174)	**26.84**⋆	**10.14**⋆	**11.14**⋆	**0.60**⋆	**0.26**⋆	**0.29**⋆	**68.72**⋆	**41.59**⋆	**5.97**⋆
KITTI06-1	ALOAM_MCL	56.90	–	–	–	–	–	–	–	–	–
	FLOAM_MCL		547.68	243.60	276.71	9.21	2.16	2.80	278.36	95.46	18.56
	KISS_MCL		–	–	–	–	–	–	–	–	–
	HDL_localization		–	–	–	–	–	–	–	–	–
	SM_MCL	5791.79	43.86	17.81	19.64	0.90	0.38	0.42	102.60	**45.58**⋆	7.35
	Range_MCL	10.80	153.86	110.38	112.82	2.89	0.75	0.95	221.04	175.73	23.96
	ERPoT(ours)	**0.27**⋆(2644/33504)	**28.78**⋆	**7.25**⋆	**8.08**⋆	**0.51**⋆	**0.23**⋆	**0.24**⋆	**63.79**⋆	46.55	**5.63**⋆
KITTI08-1	ALOAM_MCL	133.01	–	–	–	–	–	–	–	–	–
	FLOAM_MCL		–	–	–	–	–	–	–	–	–
	KISS_MCL		–	–	–	–	–	–	–	–	–
	HDL_localization		–	–	–	–	–	–	–	–	–
	SM_MCL	11061.94	**61.26**⋆	24.60	28.24	1.24	0.48	0.53	**62.89**⋆	**42.00**⋆	**5.23**⋆
	Range_MCL	25.14	–	–	–	–	–	–	–	–	–
	ERPoT(ours)	**0.59**⋆(5451/71972)	78.01	**16.10**⋆	**22.47**⋆	**0.96**⋆	**0.34**⋆	**0.40**⋆	67.14	45.39	5.33

TABLE V

EXPERIMENTAL RESULTS OF DIFFERENT APPROACHES USING THE NEWER COLLEGE DATASET

Dataset No.	Approach name	Prior map size (MB) ↓	Translational error (cm)			Rotational error (deg)			Runtime (ms)		
			Max ↓	Mean ↓	RMSE ↓	Max ↓	Mean ↓	RMSE ↓	Max ↓	Mean ↓	SD↓
Newer College-1	ALOAM_MCL	62.49	122.38	36.10	42.90	11.31	1.98	2.48	639.64	182.60	143.01
	FLOAM_MCL		–	–	–	–	–	–	–	–	–
	KISS_MCL		145.12	41.44	49.06	**9.24**⋆	1.93	2.44	237.92	74.90	23.81
	HDL_localization		–	–	–	–	–	–	–	–	–
	SM_MCL	4397.10	–	–	–	–	–	–	–	–	–
	Range_MCL	23.28	124.63	30.83	36.63	14.75	1.60	2.05	585.20	228.98	170.01
	ERPoT(ours)	**0.32**⋆(2749/40341)	**136.00**⋆	**18.08**⋆	**21.24**⋆	19.03	**0.88**⋆	**1.23**⋆	**66.18**⋆	**29.99**⋆	**8.68**⋆
Newer College-2	ALOAM_MCL	62.49	–	–	–	–	–	–	–	–	–
	FLOAM_MCL		–	–	–	–	–	–	–	–	–
	KISS_MCL		248.69	65.20	78.99	15.17	2.78	3.61	499.19	110.53	50.72
	HDL_localization		–	–	–	–	–	–	–	–	–
	SM_MCL	4397.10	–	–	–	–	–	–	–	–	–
	Range_MCL	23.28	254.92	45.79	62.44	12.80	2.01	2.60	585.59	422.85	158.01
	ERPoT(ours)	**0.32**⋆(2749/40341)	**210.54**⋆	**24.92**⋆	**29.47**⋆	**8.05**⋆	**1.03**⋆	**0.34**⋆	**59.30**⋆	**27.73**⋆	**7.07**⋆

TABLE VI

EXPERIMENTAL RESULTS OF DIFFERENT APPROACHES USING THE FUSIONPORTABLE DATASET(OUTDOOR)

Dataset No.	Approach name	Prior map size (MB) ↓	Translational error (cm)			Rotational error (deg)			Runtime (ms)		
			Max ↓	Mean ↓	RMSE ↓	Max ↓	Mean ↓	RMSE ↓	Max ↓	Mean ↓	SD↓
Portable-campus	ALOAM_MCL	24.77	206.92	45.65	57.48	11.51	1.62	2.15	590.09	266.80	134.53
	FLOAM_MCL		197.48	65.63	75.57	16.05	2.18	3.14	292.33	144.75	82.73
	KISS_MCL		132.39	61.39	70.56	7.16	1.40	1.78	390.19	116.33	49.14
	HDL_localization		249.66	25.84	31.02	12.13	1.12	1.77	564.45	127.97	126.70
	SM_MCL	2369.31	256.00	35.08	44.02	**5.85**⋆	**0.96**⋆	1.48	**71.43**⋆	**32.71**⋆	**5.48**⋆
	Range_MCL	10.68	80.66	34.82	38.30	19.59	5.57	3.59	441.12	239.27	151.00
	ERPoT(ours)	**0.10**⋆(960/12655)	**42.38**⋆	**17.08**⋆	**19.30**⋆	6.60	1.11	**1.47**⋆	101.89	58.76	10.51
Portable-building-1	ALOAM_MCL	42.33	211.11	55.91	80.42	17.46	2.54	3.56	564.36	257.12	129.66
	FLOAM_MCL		82.48	26.82	32.35	6.91	1.56	2.00	394.81	123.75	25.10
	KISS_MCL		100.37	31.07	39.66	6.88	1.74	2.23	185.86	63.45	29.50
	HDL_localization		–	–	–	–	–	–	–	–	–
	SM_MCL	3335.99	147.95	34.63	42.14	7.37	1.22	1.70	**52.08**⋆	**28.30**⋆	7.29
	Range_MCL	10.15	103.09	41.85	50.32	12.70	1.85	2.69	100.65	47.44	8.61
	ERPoT(ours)	**0.23**⋆(2290/30488)	**49.78**⋆	**12.53**⋆	**15.10**⋆	**5.33**⋆	**0.46**⋆	**0.63**⋆	81.03	56.81	**5.33**⋆
Portable-building-2	ALOAM_MCL	42.33	–	–	–	–	–	–	–	–	–
	FLOAM_MCL		78.11	25.02	29.36	10.58	2.03	2.63	216.99	115.48	44.75
	KISS_MCL		78.40	24.06	28.42	14.48	2.14	3.01	194.57	69.01	32.71
	HDL_localization		–	–	–	–	–	–	–	–	–
	SM_MCL	3335.99	–	–	–	–	–	–	–	–	–
	Range_MCL	10.15	187.95	53.31	73.61	13.07	2.51	3.40	347.70	259.41	130.56
	ERPoT(ours)	**0.23**⋆(2290/30488)	**38.15**⋆	**11.92**⋆	**13.06**⋆	**3.06**⋆	**0.61**⋆	**0.79**⋆	**95.26**⋆	**59.26**⋆	**9.23**⋆

TABLE VII

EXPERIMENTAL RESULTS OF DIFFERENT APPROACHES USING THE FUSIONPORTABLE DATASET(INDOOR)

Dataset No.	Approach name	Prior map size (MB) ↓	Translational error (cm)			Rotational error (deg)			Runtime (ms)		
			Max ↓	Mean ↓	RMSE ↓	Max ↓	Mean ↓	RMSE ↓	Max ↓	Mean ↓	SD↓
Portable-garden	ALOAM_MCL	10.53	–	–	–	–	–	–	–	–	–
	FLOAM_MCL		99.01	20.97	23.64	10.76	1.52	2.02	203.81	122.94	10.76
	KISS_MCL		168.52	18.85	26.65	16.03	1.26	1.74	159.38	65.91	17.54
	HDL_localization		–	–	–	–	–	–	–	–	–
	SM_MCL	644.31	–	–	–	–	–	–	–	–	–
	Range_MCL	16.11	**46.74**⋆	20.13	21.85	8.32	1.38	1.80	485.29	398.88	166.04
	ERPoT(ours)	**0.06**⋆(627/8135)	73.03	**15.39**⋆	**17.72**⋆	**4.31**⋆	**0.66**⋆	**0.90**⋆	**103.98**⋆	**64.34**⋆	**9.45**⋆
Portable-canteen	ALOAM_MCL	13.72	–	–	–	–	–	–	–	–	–
	FLOAM_MCL		81.12	19.09	23.10	9.05	1.49	1.95	322.74	113.17	17.34
	KISS_MCL		71.18	18.98	24.29	5.78	1.46	1.82	154.35	**55.63**⋆	13.18
	HDL_localization		–	–	–	–	–	–	–	–	–
	SM_MCL	885.53	–	–	–	–	–	–	–	–	–
	Range_MCL	18.46	**37.39**⋆	13.32	14.93	8.19	1.31	1.73	588.19	486.95	82.51
	ERPoT(ours)	**0.11**⋆(1115/15195)	43.05	**9.17**⋆	**10.22**⋆	**5.02**⋆	**0.56**⋆	**0.79**⋆	**99.08**⋆	62.45	**11.16**⋆

dataset-01 and Self dataset-02, the ground truth is derived using LIO-SAM [5], which integrates data from the LiDAR sensor, high-precision GPS, and an inertial measurement unit (IMU). This integration yields the ground truth for keyframes and records the relative pose of each frame with respect to its corresponding keyframe. Finally, linear interpolation is applied to calculate the pose of each frame, thereby obtaining the ground truth for the entire dataset.

For the Self dataset-01, the corresponding prior polygon map is shown in Fig. 20(a). There are three test sequences for evaluation of pose tracking, and the comparative experimental results are shown in Tab. VIII. Note that the first two test sequences are collected at the same time as the dataset used for constructing the prior map, while the third test sequence (Self dataset-01-3) is obtained a year and a half later. As the detailed comparative experimental results in Tab. VIII, compared to the other six approaches, the proposed ERPoT achieves the superior performance in terms of map size, translational error, and rotational error, while simultaneously maintaining high and stable algorithm running efficiency. In particular, the experimental results using Self dataset-01-3 demonstrate that the novel polygon map and pose tracking approach have long-term viability, and the long-term environmental changes in the testing environment are illustrated in Fig. 21. In contrast, HDL_localization and SM_MCL have encountered failures within this evaluation. Unlike ALOAM_MCL, FLOAM_MCL, and KISS_MCL, these methods rely solely on the prior point cloud map obtained over a year and a half ago. The primary reason for these shortcomings is the significant environmental changes that hinder point cloud registration, thereby affecting the stability and accuracy of pose tracking.

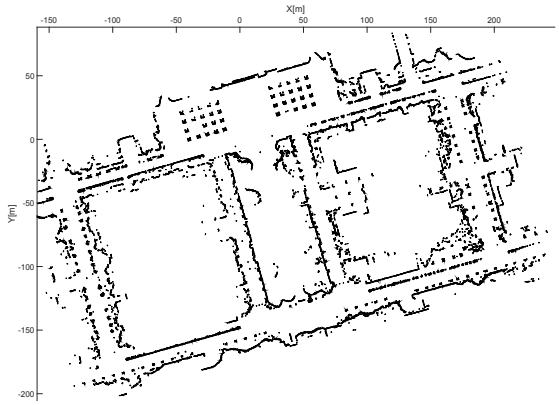
On the other hand, taking into account the diversity of real-world operating environments, the Self dataset-02 is obtained by selecting an open square that includes uphill and downhill sections. This choice ensures a comprehensive evaluation of the performance of ERPoT across diverse terrains, and simulates the challenges encountered in actual deployments. The obtained prior polygon map is shown in Fig. 20(b).

More importantly, the comparative experimental results using Self dataset-02 are shown in Tab. VIII. In particular, the prior map is remarkably compact, with a size of merely 0.65 MB, and is significantly more concise than the point cloud map and distance field representation. In terms of translational and rotational errors, the proposed ERPoT has achieved optimal or near-optimal performance, all the while maintaining high and stable operational efficiency. Experimental results demonstrate the effectiveness of ERPoT across diverse terrains.

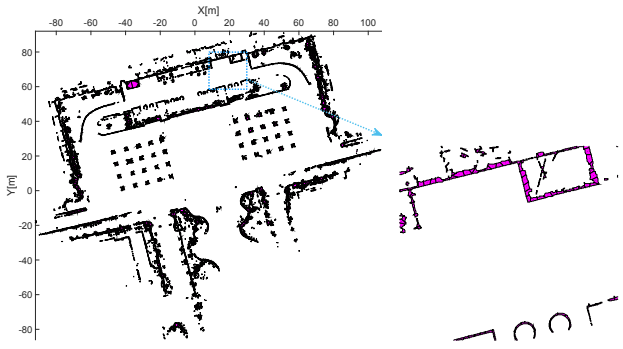
TABLE VIII

EXPERIMENTAL RESULTS OF DIFFERENT APPROACHES USING THE SELF DATASET

Dataset No.	Approach name	Prior map size (MB) ↓	Translational error (cm)			Rotational error (deg)			Runtime (ms)		
			Max ↓	Mean ↓	RMSE ↓	Max ↓	Mean ↓	RMSE ↓	Max ↓	Mean ↓	SD↓
Self dataset-01-1	ALOAM_MCL	111.91	53.49	24.30	26.08	5.59	0.91	1.25	508.59	106.54	106.63
	FLOAM_MCL		109.57	33.12	40.95	5.16	0.92	1.23	234.67	117.58	16.41
	KISS_MCL		87.40	27.74	32.71	5.48	0.90	1.19	271.03	**42.96**⋆	12.83
	HDL_localization		–	–	–	–	–	–	–	–	–
	SM_MCL	19257.15	50.42	20.96	23.13	**1.08**⋆	0.40	0.42	139.30	64.38	10.90
	Range_MCL	56.15	73.26	29.85	32.44	9.00	0.88	1.62	600.22	537.70	127.39
	ERPoT(ours)	**0.98**⋆(8619/122066)	**22.26**⋆	**9.48**⋆	**10.24**⋆	1.15	**0.11**⋆	**0.18**⋆	**81.16**⋆	50.19	**5.99**⋆
Self dataset-01-2	ALOAM_MCL	111.91	27.28	13.14	14.03	5.04	0.60	0.82	291.12	160.07	79.91
	FLOAM_MCL		42.27	14.44	15.35	2.66	0.51	0.65	265.13	100.98	14.38
	KISS_MCL		37.07	22.11	23.59	2.91	0.61	0.78	**73.78**⋆	**37.00**⋆	9.94
	HDL_localization		**21.03**⋆	14.05	14.33	1.94	1.39	1.41	768.93	55.18	25.72
	SM_MCL	19257.15	39.41	20.26	21.29	1.11	0.62	0.64	92.78	54.32	10.26
	Range_MCL	56.15	73.23	29.96	30.87	2.73	0.70	0.89	610.77	468.34	63.05
	ERPoT(ours)	**0.98**⋆(8619/122066)	28.83	**6.97**⋆	**7.99**⋆	**0.65**⋆	**0.17**⋆	**0.23**⋆	77.04	51.93	**5.03**⋆
Self dataset-01-3	ALOAM_MCL	111.91	**81.99**⋆	22.77	26.03	3.72	0.65	0.84	1030.19	199.14	200.79
	FLOAM_MCL		106.90	26.90	33.07	11.46	0.77	1.13	308.44	147.07	38.18
	KISS_MCL		–	–	–	–	–	–	–	–	–
	HDL_localization		–	–	–	–	–	–	–	–	–
	SM_MCL	19257.15	–	–	–	–	–	–	–	–	–
	Range_MCL	56.15	301.81	37.34	52.83	26.76	1.56	3.39	724.46	421.64	210.58
	ERPoT(ours)	**0.98**⋆(8619/122066)	139.96	**8.92**⋆	**10.86**⋆	**2.59**⋆	**0.14**⋆	**0.23**⋆	**108.46**⋆	**62.23**⋆	**7.53**⋆
Self dataset-02-1	ALOAM_MCL	32.92	102.32	23.09	30.57	3.13	0.64	0.83	367.93	139.53	95.97
	FLOAM_MCL		53.06	16.47	20.15	2.61	0.57	0.73	139.14	82.32	16.74
	KISS_MCL		92.98	20.29	26.28	3.30	0.63	0.83	**53.71**⋆	**31.62**⋆	**6.55**⋆
	HDL_localization		50.67	**4.81**⋆	**6.18**⋆	4.12	1.38	1.54	205.48	42.77	18.76
	SM_MCL	5589.06	30.79	11.89	13.17	**1.26**⋆	0.35	0.42	180.10	45.12	11.24
	Range_MCL	28.83	350.34	131.15	154.27	10.39	1.56	2.41	803.37	457.29	209.51
	ERPoT(ours)	**0.65**⋆(5855/78864)	**29.62**⋆	6.33	7.55	1.40	**0.23**⋆	**0.33**⋆	61.26	39.07	7.92



(a) Self dataset-01



(b) Self dataset-02

Fig. 20. Polygon maps for the Self dataset.

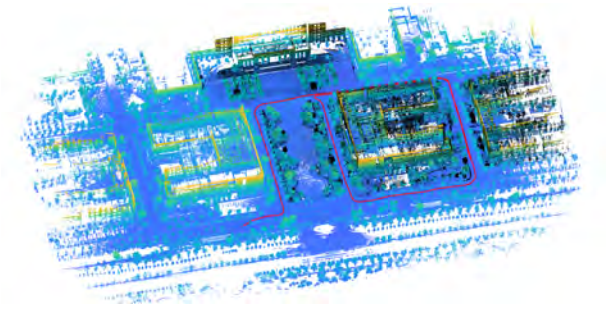


Fig. 21. Diagram of point cloud changes in the testing environment for Self dataset-01. The colored point cloud represents the prior map obtained a year and a half ago, while the black point cloud represents the point cloud information of the current testing environment.

G. Ablation study

To systematically evaluate the contributions of various components in our proposed ERPoT, an ablation study is conducted. This study aims to provide insights into how each component affects the overall performance of the system, particularly focusing on the point-vertex and point-edge constraints. Importantly, we evaluate the performance of ERPoT both with and without these constraints across all datasets.

The results of the ablation study are presented in Tab. IX. The top three columns represent translational errors, while the bottom three columns show rotational errors. Results that are worse than those of the complete ERPoT are highlighted with gray shading. It is clear that the point-edge constraint plays a significant role in successful pose tracking, and incorporating the point-edge constraint can further enhance the accuracy.

V. DISCUSSION

A. Limitations and future work

While ERPoT achieves excellent performance in map size, translational error, rotational error, and runtime, it has limitations. Specifically, ERPoT relies on ground segmentation to transform dense 3D LiDAR point clouds into sparse 2D

TABLE IX

EXPERIMENTAL RESULTS OF ABLATION STUDY FOR ERPOT WITHOUT(W/O) POINT-VERTEX CONSTRAINT OR POINT-EDGE CONSTRAINT

Dataset	No.	w/o Point-vertex constrait				w/o Point-edge constrait			
		Succ.	Impact on pose tracking			Succ.	Impact on pose tracking		
KITTI	00-1	✔	63.22	11.52	14.79	✔	48.88	16.10	19.06
			0.84	0.37	0.42		0.87	0.40	0.45
	00-2	✔	20.87	7.68	8.77	✔	33.17	10.78	12.21
			0.83	0.21	0.26		1.28	0.24	0.30
	00-3	✔	43.38	13.89	18.25	✔	38.19	15.95	19.37
			0.58	0.16	18.25		0.55	0.16	0.20
	02-1	✔	35.54	12.30	14.11	✔	30.36	12.03	13.75
			0.50	0.25	0.28		0.70	0.23	0.26
	02-2	✔	86.79	19.31	24.22	✔	77.25	19.15	22.54
			0.31	0.39	24.22		0.86	0.29	0.35
	05-1	✔	29.83	7.30	8.67	✔	28.38	11.07	12.20
			1.46	8.67	0.31		1.58	0.26	0.36
	05-2	✔	42.80	8.54	10.11	✔	36.80	14.33	15.76
			0.76	0.25	0.29		0.77	0.28	0.32
	06-1	✔	44.95	7.20	8.39	✔	26.31	9.06	9.97
			0.57	0.24	0.26		0.52	0.23	0.26
	08-1	✘				✔	72.36	17.88	23.57
							1.29	0.33	0.40
Newer College	00-1	✘				✔	146.41	19.56	22.70
							18.20	0.87	1.19
	00-2	✘				✘			
FusionPortable	campus	✘				✔	46.23	46.23	20.47
							6.58	1.01	1.39
	building-1	✘				✔	51.66	14.41	16.90
							5.59	0.46	0.64
	building-2	✘				✔	46.27	13.69	14.93
							3.15	0.63	0.81
	garden	✘				✔	266.39	17.03	27.72
							21.90	0.74	1.48
	canteen	✘				✔	47.00	11.32	12.53
							5.26	0.61	0.85
Self dataset	01-1	✔	20.29	7.36	7.86	✔	24.70	10.95	11.89
			1.51	0.10	0.18		1.13	0.12	0.19
	01-2	✔	159.53	6.17	14.24	✔	71.73	9.42	11.08
			0.85	6.17	0.22		0.68	0.18	0.24
	01-3	✘				✔	106.83	10.16	12.20
							2.55	0.15	0.24
	02-1	✘				✔	31.39	7.44	8.59
							1.13	0.23	0.33

scans. This dependency can cause issues in environments where reliable ground information cannot be extracted, such as in scenarios with uneven terrain or cluttered scenes with low obstacles. In these cases, the ground segmentation process may fail, leading to degraded performance.

Moreover, the dependency of ERPoT on ground segmentation makes it less versatile compared to other pose-tracking methods that do not require such preprocessing steps. To address these limitations, future work could explore alternative preprocessing techniques that are more robust to environmental variations. On the other hand, it is worth noting that there are alternative approaches (*e.g.*, ROS packages like pointcloud_to_laserscan) that can extract a 2D scan from a 3D LiDAR without relying on ground detection. Additionally, the lightweight and compact polygon map is ideal for multi-robot collaborative tasks, enabling efficient map sharing, faster communication, and enhanced system performance, while its concise representation supports rapid scene recognition, crucial for real-time applications like search and rescue or dynamic industrial settings.

B. Probabilistic model extension

In the field of robot pose tracking, Monte Carlo localization-based approaches play the crucial role, which makes use of a probabilistic estimation technique to properly handle uncertainties about initial, motion process, and sensor observation. In this section, we explore the integration of our proposed approach as an measurement model within the Monte Carlo Localization (MCL) framework. Our proposed ERPoT is a approach which relies on a lightweight and compact polygon map for pose tracking. Specifically, it converts dense 3D LiDAR point clouds into sparse 2D scans and employs a novel cost function for pose estimation through point-polygon matching. This approach ensures efficient and accurate pose tracking, making it suitable for integration with MCL.

The complete probabilistic localization process contains initialization, motion model, measurement model, resampling and pose estimation. In this section, we focus on the measurement model, which makes use of the kernel contributions on the proposed novel polygon map representation and the data association between the point and the polygon. For time stamp t, given the generated sparse 2D scan observation $\mathbf{C}_t$ and the lightweight polygon map $\mathcal{M}_p$, the measurement model is represented as follows

$$p(\mathbf{C}_t^{hit}|\tilde{\mathbf{s}}_t, \mathcal{M}_p) = \prod_{i=1}^{K} p\left(\mathbf{p}_i^{2D}|\tilde{\mathbf{s}}_t, \mathbf{P}_{i\to j}\right), \tag{31}$$

where the j-th corresponding polygon $\mathbf{P}_j \in \mathcal{M}_p$, and $\mathbf{C}_t^{hit}$ is subset of $\mathbf{C}_t$ with K scan points that meet the condition

$$d^*(\mathbf{p}_i^{2D}, \mathcal{M}_p) = \min_{j=1,2,\ldots,m} d\left(\mathbf{p}_i^{2D}, \mathbf{P}_j\right) < \delta_d, \tag{32}$$

and δ_d is the data association threshold. Accordingly,

$$p\left(\mathbf{p}_i^{2D}|\tilde{\mathbf{s}}_t, \mathbf{P}_{i\to j}\right) = \mathcal{N}\left(d^*; 0, \sigma_{hit}^2\right), \tag{33}$$

where σ_{hit} is the standard deviation.

VI. CONCLUSION

In this study, an efficient and reliable pose tracking framework for mobile robots called ERPoT has been proposed, which makes use of a novel prior map constructed from multiple compact polygons. Notably, this system operates solely on LiDAR data, acquiring the sparse 2D scan points through ground removal and obstacle selection. Following this, an offline map construction procedure is executed, yielding an environment map that is both lightweight and compact in its polygon representation. Ultimately, by integrating real-time 2D scan points with the prior polygon map, our approach achieves reliable long-term pose tracking based on the newly proposed point-polygon matching. Furthermore, the quantitative evaluation shows the superior performance on prior map size, pose estimation error, and runtime for pose tracking compared with the other six approaches.

REFERENCES

[1] J. Placed, J. Strader, H. Carrillo, *et al.*, "A survey on active simultaneous localization and mapping: State of the art and new frontiers," *IEEE Trans. Robot.*, vol. 39, no. 3, pp. 1686-1705, 2023.

[2] J. Wen, X. Zhang, H. Gao, *et al.*, "E3MoP: Efficient motion planning based on heuristic-guided motion primitives pruning and path optimization with sparse-banded structure," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 4, pp. 2762-2775, 2022.

[3] P. Singamaneni, P. Bachiller-Burgos, L. Manso, *et al.*, "A survey on socially aware robot navigation: Taxonomy and future challenges," *Int. J. Robot. Res.*, vol. 43, no. 10, pp. 1533-1572, 2024.

[4] C. Campos, R. Elvira, J. Rodríguez, *et al.*, "ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap slam," *IEEE Trans. Robot.*, vol. 37, no. 6, pp. 1874-1890, 2021.

[5] T. Shan, B. Englot, D. Meyers, *et al.*, "LIO-SAM: Tightly-coupled lidar inertial odometry via smoothing and mapping," In *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 5135-5142.

[6] J. Deng, Q. Wu, X. Chen, *et al.*, "NeRF-LOAM: Neural implicit representation for large-scale incremental lidar odometry and mapping," In *IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 8218-8227.

[7] F. Graf, J. Lindermayr, B. Graf, *et al.*, "HIPer: A human-inspired scene perception model for multifunctional mobile robots," *IEEE Trans. Robot.*, vol. 40, pp. 4668-4683, 2024.

[8] C. Wang, X. Chen, C. Li, *et al.*, "Chase and track: Toward safe and smooth trajectory planning for robotic navigation in dynamic environments," *IEEE Trans. Ind. Electron.*, vol. 70, no. 1, pp. 604-613, 2023.

[9] Q. Bi, X. Zhang, J. Wen, *et al.*, "CURE: A hierarchical framework for multi-robot autonomous exploration inspired by centroids of unknown regions," *IEEE Trans. Autom. Sci. Eng.*, vol. 21, no. 3, pp. 3773-3786, 2024.

[10] H. Yin, X. Xu, S. Lu, *et al.*, "A survey on global lidar localization: Challenges, advances and open problems," *Int. J. Comput. Vis.*, vol. 132, pp. 3139-3171, 2024.

[11] K. Koide, M. Yokozuka, S. Oishi, *et al.*, "Voxelized GICP for fast and accurate 3D point cloud registration," In *Proc. IEEE Int. Conf. Robot. Auton.*, 2021, pp. 11054-11059.

[12] R. Dellenbach, J. Deschaud, B. Jacquet, *et al.*, "CT-ICP: Real-time elastic lidar odometry with loop closure," In *Proc. IEEE Int. Conf. Robot. Auton.*, 2022, pp. 5580-5586.

[13] I. Vizzo, T. Guadagnino, B. Mersch, *et al.*, "KISS-ICP: In defense of point-to-point icp–simple, accurate, and robust registration if done the right way," *IEEE Robot. Autom. Lett.*, vol. 8, no. 2, pp. 1029-1036, 2023.

[14] J. Zhang and S. Singh, "LOAM: Lidar odometry and mapping in real-time," In *Robot.: Sci. Syst.*, vol. 2, no. 9, pp. 1-9, 2014.

[15] T. Shan and B. Englot, "LeGO-LOAM: Lightweight and ground-optimized lidar odometry and mapping on variable terrain," In *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 4758-4765.

[16] H. Dong, X. Chen, S. Särkkä, *et al.*, "Online pole segmentation on range images for long-term LiDAR localization in urban environments," *Robot. Auton. Syst.*, vol. 159, pp. 104283, 2023.

[17] S. Isaacson, P. Kung, M. Ramanagopal, *et al.*, "LONER: LiDAR only neural representations for real-time slam," *IEEE Robot. Autom. Lett.*, vol. 8, no. 12, pp. 8042-8049, 2023.

[18] R. Huang, M. Zhao, J. Chen, *et al.*, "KDD-LOAM: Jointly learned keypoint detector and descriptors assisted LiDAR odometry and mapping," In *Proc. IEEE Int. Conf. Robot. Auton.*, 2024, pp. 8559-8565.

[19] X. Chen, I. Vizzo, T. Läbe, *et al.*, "Range image-based LiDAR localization for autonomous vehicles," In *Proc. IEEE Int. Conf. Robot. Auton.*, 2021, pp. 5802-5808.

[20] N. Akai, "Efficient solution to 3D-LiDAR-based monte carlo localization with fusion of measurement model optimization via importance sampling," In *Proc. IEEE/SICE Int. Symp. Syst. Integr.*, 2025, pp. 1247-1254.

[21] F. Rico, J. Hernández, R. Pérez-Rodríguez, *et al.*, "Open source robot localization for nonplanar environments," *J. Field Robot.*, vol. 41, no. 6, pp. 1922-1939, 2024.

[22] R. Dubé, A. Cramariuc, D. Dugas, *et al.*, "SegMap: Segment-based mapping and localization using data-driven descriptors," *Int. J. Robot. Res.*, vol. 39, no. 2-3, pp. 339-355, 2020.

[23] Y. Huang, Y. Gu, C. Xu, *et al.*, "Why semantics matters: A deep study on semantic particle-filtering localization in a LiDAR semantic pole-map," *IEEE Trans. Field Robot.*, vol. 1, pp. 47-69, 2024.

[24] J. Kümmerle, M. Sons, F. Poggenhans, *et al.*, "Accurate and efficient self-localization on roads using basic geometric primitives," In *Proc. IEEE Int. Conf. Robot. Auton.*, 2019, pp. 5965-5971.

[25] D. Yu, Y. Hu, Y. Li, *et al.*, "PolygonGNN: Representation learning for polygonal geometries with heterogeneous visibility graph," In *KDD*, 2024, pp. 4012-4022.

[26] S. Ferrari, L. Di Giammarino, L. Brizi, *et al.*, "MAD-ICP: It is all about matching data–robust and informed LiDAR odometry," *IEEE Robot. Autom. Lett.*, vol. 9, no. 11, pp. 9175-9182, 2024.

[27] W. Xu, Y. Cai, D. He, *et al.*, "Fast-LIO2: Fast direct lidar-inertial odometry," *IEEE Trans. Robot.*, vol. 38, no. 4, pp. 2053-2073, 2022.

[28] H. Wang, C. Wang, C. Chen, *et al.*, "F-LOAM: Fast lidar odometry and mapping," In *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 4390-4396.

[29] S. Yi, Y. Lyu, L. Hua, *et al.*, "Light-LOAM: A lightweight LiDAR odometry and mapping based on graph-matching," *IEEE Robot. Autom. Lett.*, vol. 9, no. 4, pp. 3219-3226, 2024.

[30] Z. Qin, H. Yu, C. Wang, *et al.*, "Geotransformer: Fast and robust point cloud registration with geometric transformer," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 8, pp. 9806-9821, 2023.

[31] P. Egger, P. Borges, G. Catt, *et al.*, "Posemap: Lifelong, multi-environment 3d lidar localization," In *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 3430-3437.

[32] Y. Feng, Z. Jiang, Y. Shi, *et al.*, "Block-map-based localization in large-scale environment,". In *Proc. IEEE Int. Conf. Robot. Auton.*, 2024, pp. 1709-1715.

[33] F. Xie and S. Schwertfeger, "Robust lifelong indoor lidar localization using the area graph," *IEEE Robot. Autom. Lett.*, vol. 9, no. 1, pp. 531-538, 2024.

[34] R. Dubé, A. Cramariuc, D. Dugas, *et al.*, "SegMap: 3D segment mapping using data-driven descriptors," arXiv preprint arXiv:1804.09557, 2018.

[35] R. Dubé, M. Gollub, H. Sommer, *et al.*, "Incremental-segment-based localization in 3-D point clouds," *IEEE Robot. Autom. Lett.*, vol. 3, no. 3, pp. 1832-1839, 2018.

[36] A. Schaefer, D. Büscher, J. Vertens, *et al.*, "Long-term vehicle localization in urban environments based on pole landmarks extracted from 3-D lidar scans," *Robot. Auton. Syst.*, vol. 136, pp. 103709, 2021.

[37] O. Hafez, M. Joerger, and M. Spenko, "How safe is particle filtering-based localization for mobile mobots? An integrity monitoring approach," *IEEE Trans. Robot.*, vol. 40, pp. 3372-3387, 2024.

[38] R. Wolcott and R. Eustice, "Robust LiDAR localization using multiresolution Gaussian mixture maps for autonomous driving," *Int. J. Robot. Res.*, vol. 36, no. 3, pp. 292-319, 2017.

[39] K. Koide, J. Miura, and E. Menegatti, "A portable three-dimensional LIDAR-based system for long-term and wide-area people behavior measurement," *Int. J. Adv. Robot. Syst.*, vol. 16, no. 2, pp. 1-16, 2019.

[40] B. Peng, H. Xie, and W. Chen, "Roll: Long-term robust LiDAR-based localization with temporary mapping in changing environments," In *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 2841-2847.

[41] I. Hroob, S. Molina, R. Polvara, *et al.*, "Learned long-term stability scan filtering for robust robot localisation in continuously changing environments," In *European Conference on Mobile Robots*, 2023, pp. 1-8.

[42] L. Montano-Oliván, J. Placed, L. Montano, *et al.*, "G-Loc: Tightly-coupled graph localization with prior topo-metric information," *IEEE Robot. Autom. Lett.*, vol. 9, no. 11, pp. 9167-9174, 2024.

[43] K. Konolige, G. Grisetti, R. Kümmerle, *et al.*, "Efficient sparse pose adjustment for 2D mapping," In *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2010, pp. 22-29.

[44] S. Lee, H. Lim, and H. Myung, "Patchwork++: Fast and robust ground segmentation solving partial undersSegmentation using 3D point cloud," In *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 13276-13283.

[45] H. Gao, Q. Qiu, W. Hua, *et al.*, "CVR-LSE: Compact vectorized representation of local static environments for reliable obstacle detection," *IEEE Trans. Ind. Electron.*, vol. 71, no. 8, pp. 9309-9318, 2024.

[46] A. Geiger, P. Lenz, C. Stiller, *et al.*, "Vision meets robotics: The kitti dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231-1237, 2013.

[47] M. Ramezani, Y. Wang, M. Camurri, *et al.*, "The newer college dataset: Handheld lidar, inertial and vision with ground truth," In *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 4353-4360.

[48] J. Jiao, H. Wei, T. Hu, *et al.*, "Fusionportable: A multi-sensor campus-scene dataset for evaluation of localization and mapping accuracy on diverse platforms," In *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 3851-3856.

[49] Z. Zhang and D. Scaramuzza, "A tutorial on quantitative trajectory evaluation for visual (-inertial) odometry," In *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 7244-7251.