SE-LIO: Semantic-Enhanced Solid-State-LiDAR–Inertial Odometry for Tree-Rich Environments

Tisheng Zhang, *Member, IEEE*, Linfu Wei, Hailiang Tang, Man Yuan, Liqiang Wang, and Xiaoji Niu, *Member, IEEE*

***Abstract*— LiDAR–inertial odometry (LIO) based on line and plane features will face great challenges in environments abundant with unstructured features, such as tree-rich environments. In this article, we propose a semantic-enhanced solid-state-LIO (SE-LIO) in tree-rich environments. Multiple LiDAR frames are first merged and compensated with the inertial navigation system (INS) to increase the point-cloud coverage, thus improving the accuracy of semantic segmentation. Then, the unstructured point clouds, such as tree leaves and dynamic objects, are removed with the semantic information. Besides, the pole-like point clouds, primarily tree trunks, are modeled as cylinders to improve positioning accuracy. We also propose an adaptive piecewise cylinder-fitting method to accommodate environments with a high prevalence of curved tree trunks. Finally, an efficient iterated error-state Kalman filter (IESKF) is employed to integrate LiDAR and inertial measurements tightly. Exhaustive experiments are conducted on public and private datasets with complex campus and park scenes to evaluate the proposed SE-LIO. The results demonstrate that the proposed baseline LIO yields improved or comparable accuracy to the state-of-the-art methods. By incorporating the semantic enhancement, the absolute translation accuracy of the proposed SE-LIO on the public and private datasets is improved by 17.1% and 43.1%, respectively, compared to the baseline system. Besides, the ablation experiment results illustrate that the positioning accuracy is also improved by employing the proposed adaptive cylinder fitting.**

***Index Terms*— LiDAR–inertial navigation, multisensor fusion navigation, pole-like point cloud, semantics enhancement, state estimation.**

I. Introduction

CONTINUOUS, reliable, and accurate positioning in complex environments is crucial for autonomous vehicles and mobile robots. While LiDAR–inertial odometry (LIO) based on line and plane features has demonstrated excellent performance in structured environments, it encounters significant challenges in scenes abundant with unstructured features, such as tree-rich campuses and parks. Fig. 1 illustrates some typical campus and park scenes. In these scenarios, roads are surrounded by trees, resulting in a high proportion of unstructured point clouds (mainly tree leaves point clouds) in the LiDAR-scanned point cloud. These unstructured point clouds may significantly decrease the accuracy of traditional positioning methods that rely on geometric features, such as extracting plane feature points from unstructured point clouds like tree leaves. These plane feature points lack sufficient accuracy, leading to reduced positioning accuracy.

A. Related Work

Normal distribution transform (NDT)-based methods [1], [2], [3], [4] have been presented to address these problems. These methods rely on statistical features of point clouds, such as the mean and covariance, rather than geometric features. They have demonstrated commendable positioning accuracy in unstructured environments. However, the matching accuracy and efficiency of NDT-based methods are intrinsically linked to the size of the grid into which they are divided. A smaller grid size yields higher matching accuracy but compromises matching efficiency and vice versa. Adaptive voxel mapping [5] is a similar method. It divides point clouds into voxel grids, each containing an octree. The octree nodes contain the distribution information of the point clouds in the node. If the point clouds in the node do not conform to a planar distribution, the node is divided into eight subnodes, and the distribution of the point clouds is further determined in the subnodes, achieving adaptive planar fitting.

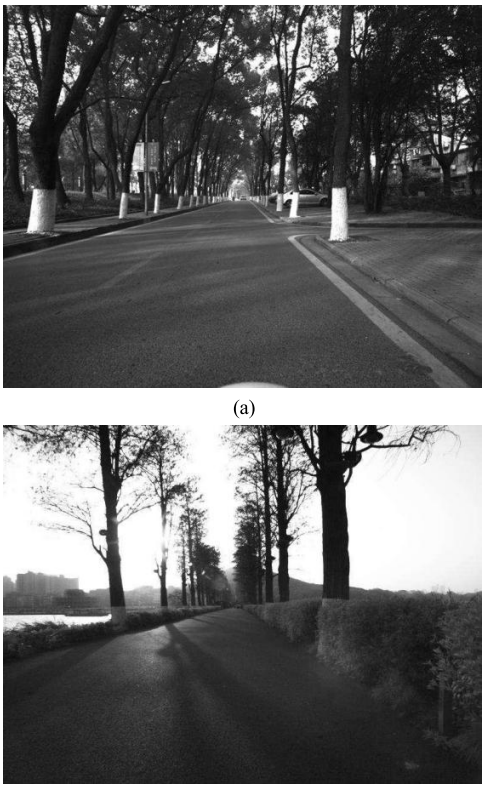
Additional methods preprocess point clouds to mitigate the impact of unstructured point clouds. For instance, LeGO-LOAM [6] segregates point clouds into ground and nonground points, further segmenting the nonground points and filtering out small nonground point-cloud clusters. It effectively reduces the influence of unstructured point clouds. The segmentation method employed is based on traditional image processing methods [7] for range images. Recently, many networks have been developed for semantic segmentation of point clouds [8]. These models can be categorized into range image-based models [9], [10], voxel-based models [11], [12], and point-based models [13], [14], [15] based on the segmentation objects.

Received 7 September 2024; revised 10 November 2024; accepted 5 December 2024. Date of publication 5 March 2025; date of current version 19 March 2025. This work was supported in part by the National Natural Science Foundation of China under Grant 42374034, in part by the Key Research and Development Program of Hubei Province under Grant 2024BAB024, in part by the Major Program (JD) of Hubei Province under Grant 2023BAA02602, and in part by the High Quality Development Project of the Ministry of Industry and Information Technology (MIIT) under Grant 2024-182. The Associate Editor coordinating the review process was Dr. Yan Zhuang. *(Corresponding author: Hailiang Tang.)*

Tisheng Zhang and Xiaoji Niu are with the GNSS Research Center and Hubei Luojia Laboratory, Wuhan University, Wuhan 430079, China (e-mail: zts@whu.edu.cn; xjniu@whu.edu.cn).

Linfu Wei, Hailiang Tang, Man Yuan, and Liqiang Wang are with the GNSS Research Center, Wuhan University, Wuhan 430079, China (e-mail: weilf@whu.edu.cn; thl@whu.edu.cn; yuanman@whu.edu.cn; wlq@whu.edu.cn).

Digital Object Identifier 10.1109/TIM.2025.3548196



(a) (b)

Fig. 1. Typical tree-rich scenes abundant with unstructured features. (a) Campus. (b) Park.

As research progresses, the point-cloud semantic segmentation accuracy has improved, effectively distinguishing between different types of point clouds, such as ground, buildings, tree trunks, tree leaves, and vehicles. This semantic information can be used to enhance positioning. Several studies have applied semantic segmentation to LiDAR positioning. SuMa++ [16] uses semantic segmentation results to weight feature points, reducing the impact of dynamic objects. PSF-LO [17] performs geometric modeling on several types of semantic segmentation results types, modeling roads, buildings, and traffic signs as planar features and poles as line features. SLOAM [18] is designed for tree-rich scenes, modeling tree trunks as cylindrical features, but it does not consider the curvature of tree trunks and is therefore not suitable for curved tree-rich environments. Moreover, the cylinder-fitting and semantic segmentation methods used in SLOAM are suitable for spinning LiDAR, but not solid-state LiDAR. SA-LOAM [19] uses semantic information for front-end odometry and loop detection, reducing incorrect matches using semantic label constraints. The research mentioned above, which utilizes semantic information to enhance positioning, is mainly designed for spinning LiDAR. It involves projecting the point clouds into a range image for semantic segmentation, which is unsuitable for solid-state LiDAR. The detailed reason will be presented in Section III-A.

Recently, the low-cost solid-state LiDARs have been widely used in autonomous robots [20], [21]. Unlike spinning LiDARs, solid-state LiDARs encompass several technologies including micro-electromechanical system (MEMS)-based, flash, and nonrepetitive scanning systems. We focus on nonrepetitive-scanning-based LiDAR, which produces distinct sampling characteristics. A key feature of such systems is their temporal integration property—the point-cloud coverage increases progressively with scanning time. The scanning pattern creates spatially varying point distributions, with significantly lower point density at the edges of field of view compared to central regions. While spinning LiDARs achieve uniform spatial sampling through consistent rotation, this nonuniform sampling pattern in nonrepetitive-scanning-based LiDAR affects the quality and consistency of range image generation. Therefore, to better accommodate these unique scanning characteristics and optimize semantic segmentation performance, a point-based semantic segmentation model is more appropriate than traditional range image-based approaches. However, the point cloud obtained by a single frame (generally 0.1 s) of solid-state LiDAR is relatively sparse and has low point-cloud coverage [22]. This sparsity often results in missed point clouds from many objects, decreasing semantic segmentation accuracy. While the point-cloud coverage can be increased by extending the scanning cycle of a single-frame point cloud when the carrier is stationary, motion distortion occurs in the point cloud when the carrier is in motion. The longer the scanning cycle, the more notable the effect of motion distortion, which is detrimental to semantic segmentation. With the development of multisource fusion technology, inertial measurement unit (IMU) has become standard equipment for LiDAR positioning and has been used to compensate for motion distortion [23]. Furthermore, IMU information is tightly coupled with LiDAR information to improve positioning accuracy and robustness [24], [25], [26]. Therefore, IMU information can be used to compensate for motion distortion, improving the point-cloud coverage and, consequently, the semantic segmentation accuracy.

B. Contribution

We propose a semantic-enhanced solid-state-LIO (SE-LIO) for scenes abundant with unstructured features. The proposed method leverages an inertial navigation system (INS) pose to merge and compensate for multiple LiDAR frames. A deep learning model is employed for semantic segmentation. The segmentation results are then utilized to remove unstructured point clouds and incorporate cylindrical features into state estimation, enhancing positioning accuracy. The primary contributions are as follows.

1) To address the low semantic segmentation accuracy caused by sparse point clouds of the solid-state LiDAR, we design an INS-enhanced semantic segmentation method, which leverages the INS pose to merge and compensate for multiple LiDAR frames, thereby improving the point-cloud coverage and semantic segmentation accuracy.
2) To fully leverage pole-like semantic information, we propose an adaptive piecewise cylinder-fitting method, effectively accommodating environments with curved trees, thereby enhancing the system's environmental adaptability and positioning accuracy.

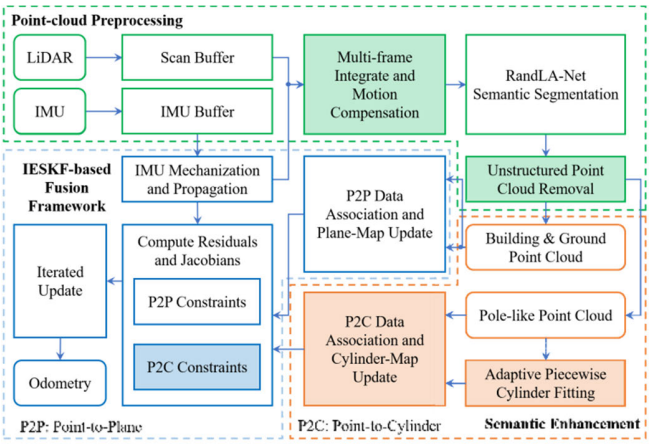


Fig. 2. System overview of the proposed SE-LIO. The proposed baseline LIO can be obtained by removing the semantic enhancement and treating all point clouds as plane points.

3) We introduce a cylinder measurement model into the iterated error-state Kalman filter (IESKF) tightly coupled state estimation framework based on plane measurements, effectively improving positioning accuracy.

We verify the accuracy and robustness of the proposed method on the public and private datasets with complex campus and park scenes. Several ablation experiments were carried out to fully evaluate the impacts of the factors that may influence the accuracy of the proposed SE-LIO.

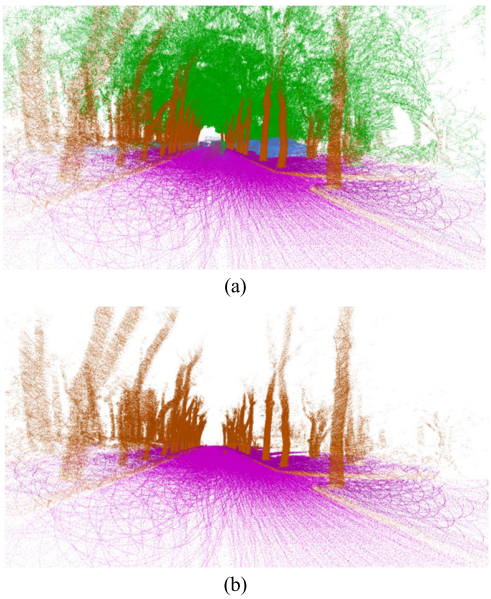
The remainder of this article is organized as follows. We give an overview of the system pipeline in Section II. The proposed method is presented in Section III. The experiments and results for quantitative evaluation are discussed in Section IV. Finally, we conclude the proposed method.

II. System Overview

The system workflow is illustrated in Fig. 2. The point clouds and IMU data are first accumulated until a certain threshold is reached. The current pose is then propagated forward using the IMU mechanization, and the point clouds undergo motion compensation. Following this, the motion-compensated point clouds are subjected to semantic segmentation, dividing them into different types, including ground, pole-like, building, tree leaves, and dynamic objects. The unstructured point clouds, primarily tree leaves and dynamic objects, are removed to mitigate their impacts on positioning accuracy. Subsequently, the pole-like semantic information is leveraged to enhance positioning accuracy, including adaptive piecewise cylinder fitting of pole-like point clouds and data association. Finally, the IESKF is employed for state estimation. Cylindrical features and plane features are used to construct point-to-cylinder and point-to-plane constraints. These constraints are tightly coupled with the prior constraints provided by INS to obtain the maximum a posteriori estimation.

III. Semantic-Enhanced Point-Cloud Processing

This section introduces the methodology of semantic-enhanced point-cloud preprocessing. It begins with the point-cloud preprocessing phase, followed by single-cylinder-fitting and adaptive piecewise cylinder-fitting methods. Finally, it presents the cylinder map updating and point-to-cylinder data association methods.



(a)

(b)

Fig. 3. Semantic segmentation results. The purple, green, and brown points represent ground, tree leaves, and pole-like point clouds, respectively. (a) Original segmentation results. (b) Removing unstructured point clouds.

A. Point-Cloud Preprocessing

The adopted LiDAR is a solid-state nonrepetitive scanning LiDAR. As mentioned in Section I-A, a point-based semantic segmentation model is more appropriate than traditional range image-based approaches. Therefore, we employ a point-based semantic segmentation method, RandLA-Net [15]. It should be noted that RandLA-Net is not necessary, and other semantic segmentation methods are also applicable to the proposed method. Given the relative sparsity of the point cloud from a single frame, we leverage the INS pose to merge and compensate for multiple LiDAR frames, thereby improving the point-cloud coverage and semantic segmentation accuracy.

Specifically, a fixed-length buffer is established to cache point cloud and IMU measurement. When the number of point clouds in the buffer reaches a predetermined threshold, the point clouds are projected to the end of the last frame of the point cloud in the buffer using INS pose, thereby generating the motion-compensated point clouds.

The motion-compensated point clouds are fed into the RandLA-Net [15] model for semantic segmentation. However, the pretrained RandLA-Net is only suitable for spinning LiDAR and not solid-state LiDAR. Therefore, we performed transfer learning to adapt RandLA-Net to solid-state LiDAR. The segmentation result is depicted in Fig. 3(a), where the purple, green, and brown points represent ground, tree leaves, and pole-like point clouds, respectively. Owing to the substantial influence of unstructured point clouds on positioning accuracy, tree leaves and dynamic objects point clouds are removed, as depicted in Fig. 3(b).

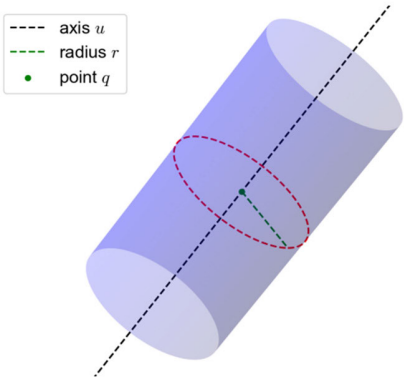


Fig. 4. Cylinder model parameterization.

After removing unstructured point clouds, the remaining point clouds predominantly consist of ground, building, and pole-like point clouds, such as tree trunks. We employ the cylinder model to fit these pole-like point clouds. Considering the curvature of pole-like objects in the environment, a piecewise fitting method adaptively fits these curved pole-like point clouds. The specific method is described in detail next.

B. Single Cylinder Fitting

A cylinder can typically be represented by a minimum of five parameters, with four parameters delineating the axis and one parameter indicating the radius. However, this minimal parameter representation lacks intuitiveness. Therefore, we select a seven-parameter representation for the cylinder model, as shown in the following equation:

$$\boldsymbol{c} = \left(\boldsymbol{u}^T, \boldsymbol{q}^T, r\right)^T \tag{1}$$

where $\boldsymbol{u}$ represents the unit vector in the direction of the axis, $\boldsymbol{q}$ represents a point on the axis, and r represents the radius of the cylinder. The schematic of the cylinder model parameterization is shown in Fig. 4.

The process of fitting a single cylinder is shown in Fig. 5. Given a cluster of pole-like point clouds obtained by semantic segmentation, let $P = \{\boldsymbol{p}\}$ represent this cluster. The procedure of fitting a cluster of pole-like point clouds to a cylinder can be summarized in the following steps.

Step 1: Estimate the axis direction $\boldsymbol{u}$. It is achieved by solving for the maximum eigenvalue of the point-cloud covariance. The corresponding eigenvector is the estimated axis vector $\boldsymbol{u}$.

Step 2: Construct the rotation matrix $\mathbf{R}$ using the axis $\boldsymbol{u}$ to transform the point cloud P, i.e., $\boldsymbol{p}' = \mathbf{R}\boldsymbol{p}$, such that the distribution of the transformed point cloud in the z-axis is maximized. The point is then projected onto the xOy plane, converting the 3-D cylinder-fitting problem into a 2-D circle fitting problem.

Step 3: Employ the random sample consensus (RANSAC) algorithm [27], [28] to calculate the circle parameters. Within RANSAC, the least-squares method fits the circle to minimize the sum of the distances between all given points and the circle. Let the circle equation be represented as

$$a_1\left(x^2 + y^2\right) + a_2 x + a_3 y + a_4 = 0 \tag{2}$$

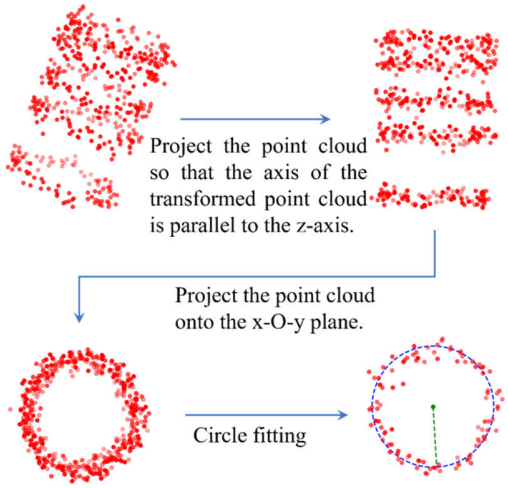


Fig. 5. Process of fitting a single cylinder.

where (x, y) represents the point coordinates on the 2-D plane, and a_1, a_2, a_3, and a_4 are the parameters of the circle. For each point in the point cloud $\mathbf{P}'$, construct the equation

$$\begin{cases} a_1\left(x_1^2 + y_1^2\right) + a_2 x_1 + a_3 y_1 + a_4 = 0 \\ a_1\left(x_2^2 + y_2^2\right) + a_2 x_2 + a_3 y_2 + a_4 = 0 \\ \qquad\qquad\qquad\qquad\qquad\qquad \vdots \\ a_1\left(x_N^2 + y_N^2\right) + a_2 x_N + a_3 y_N + a_4 = 0. \end{cases} \tag{3}$$

Express the above equations in a matrix form $\mathbf{A}\boldsymbol{a} = 0$, where $\boldsymbol{a} = (a_1, a_2, a_3, a_4)^T$, $\mathbf{A}$ is an $N \times 4$ matrix, where the ith row is $(x_i^2 + y_i^2, x_i, y_i, 1)$. Multiply both sides by $\mathbf{A}^T$ to obtain

$$\mathbf{A}^T\mathbf{A}\boldsymbol{a} = \mathbf{0}. \tag{4}$$

Then, by solving the eigenvector corresponding to the largest eigenvalue of the matrix $\mathbf{A}^T\mathbf{A}$, the parameters $\boldsymbol{a}$ of the circle can be obtained, and finally, the parameters (a_1, a_2, a_3, a_4) are converted to (x_0, y_0, r) by the following formula:

$$\begin{aligned} x_0 &= -\frac{a_2}{2a_1} \\ y_0 &= -\frac{a_3}{2a_1} \\ r &= \sqrt{x_0^2 + y_0^2 - \frac{a_4}{a_1}} \end{aligned} \tag{5}$$

where x_0 and y_0 represent the 2-D coordinates of the circle center and r represents the radius.

Step 4: Convert the circle parameters to cylinder parameters. The rotation matrix $\mathbf{R}$ is used to convert the circle center coordinates (x_0, y_0) to the point $\boldsymbol{q}$ on the cylinder axis, and the radius r is used as the cylinder radius. Finally, the parameters $\boldsymbol{c} = (\boldsymbol{u}^T, \boldsymbol{q}^T, r)^T$ of the cylinder are obtained.

It should be noted that our method implements a two-stage optimization strategy: first fitting the axis line, followed by radius optimization. This decoupled approach provides enhanced numerical stability and computational efficiency compared to single-stage methods used in systems like PLC-LiSLAM [29].

Algorithm 1 Adaptive Piecewise Cylinder Fitting

Input: Point cloud P, depth d.
Output: Tree node n.
procedure BUILD_TREE(P, d)
  if $d > D_{max}$ **then** // D_{max} represents the maximum depth of the binary tree
    return None
  end
  $C \leftarrow$ FIT_CYLINDER(P)
  if $C.\varepsilon < \varepsilon_{max}$ **then** // ε_{max} represents the maximum fitting error
    return C
  else
    // Divide the point cloud into upper and lower parts
    $P_u, P_d \leftarrow$ DIVIDE_POINT_CLOUD(P) // P_u represents the upper part, and P_d represents the lower part
    $C_u \leftarrow$ BUILD_TREE($P_u, d+1$) // C_u represents the upper child
    $C_d \leftarrow$ BUILD_TREE($P_d, d+1$) // C_d represents the lower child
  end
end

C. Adaptive Piecewise Cylinder Fitting for Pole-Like Point Clouds

To address pole-like structures with complex geometries that deviate from ideal cylindrical shapes, we propose an adaptive piecewise cylinder-fitting algorithm leveraging a binary tree structure. The method recursively segments and fits multiple cylinders while maintaining geometric continuity.

Algorithm 1 outlines the adaptive piecewise fitting process for pole-like objects. The algorithm employs a binary tree representation where each node corresponds to a cylinder segment. Parent nodes encompass complete point-cloud segments, while child nodes represent subdivisions. The tree depth dynamically adapts to local geometric variations, with a maximum depth $D_{\max}$ constraint to ensure computational efficiency.

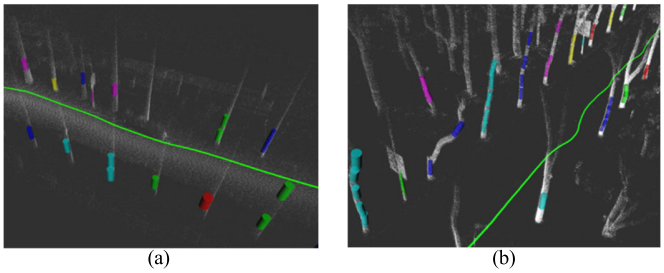
The segmentation process begins with fitting a single cylinder to the entire pole segment. The quality of fit is evaluated using the root mean square (rms) error ε compared against a threshold $\varepsilon_{\max} = 0.05$ m. When $\varepsilon > \varepsilon_{\max}$, the point cloud is bisected at the median height, creating two child segments. This recursive refinement continues until one of two termination criteria is met: 1) $\varepsilon \leq \varepsilon_{\max}$ and 2) maximum tree depth $D_{\max}$ reached.

D. Cylinder Map Update

Given limited pole-like objects, the efficiency of the algorithm will not be significantly impacted even if a linear search is employed for matching. Therefore, we use a linear array to store all trees. A coarse matching method is employed to match a new frame of pole-like point clouds with the pole-like objects on the map. If the match is successful, the pole-like object in the map is updated; otherwise, the point does not belong to any tree in the map.

Algorithm 2 Cylinder Map Update

Input: Point cloud P, Cylinder map M.
Output: Updated Cylinder map M.
procedure UPDATE_MAP(P, M)
  if not initialized **then**
    add P to buffer and return **if** buffer is not full
    initialized $\leftarrow$ true
  add P to buffer
  for each $\boldsymbol{p}$ in P **do**
    if $\boldsymbol{p}$ does not belong to any tree **then**
      if enough nearest points around $\boldsymbol{p}$ **then**
        create new tree from cluster and add to M
      end
    else
      mark $\boldsymbol{p}$ as a point to update
    end
  end
  UPDATE_TREES_IN_MAP(M)
  DELETE_OLD_POINTS_IN_BUFFER()
end



(a) (b)

Fig. 6. Cylinder-fitting results. The tree trunks that do not fit are due to the distance being too far and the point cloud being too sparse to fit a cylinder. (a) Straight pole-like point cloud. (b) Curved pole-like point cloud.

We employ the density-based spatial clustering of applications with noise (DBSCAN) [30] algorithm to cluster these new points. The map-update algorithm is shown in Algorithm 2. It was employed to fit cylinders and update the cylinder map using the campus dataset, yielding the results shown in Fig. 6. As shown in Fig. 6, the proposed method can fit cylinders to straight and curved pole-like objects. For curved pole-like objects, the proposed method can adaptively fit them in segments, rather than using a single-cylinder model.

E. Point-to-Cylinder Data Association

The matching process employs a coarse-to-fine matching strategy, as shown in Algorithm 3. The nearest tree to the current point is identified via a linear nearest neighbor search. If the distance between the current point and the nearest tree is less than the predetermined threshold, the point is considered part of that tree. Conversely, if the distance exceeds the threshold, the point is deemed independent of any tree. After confirming the tree to which the point belongs, the specific cylinder to which the point belongs on the tree is determined using binary search.

Algorithm 3 Point-to-Cylinder Data Association

Input: point $\boldsymbol{p}$, cylinder map M.
Output: cylinder C that point $\boldsymbol{p}$ belongs to.
procedure ASSOCIATE_POINT_TO_CYLINDER($\boldsymbol{p}$, M)
  // Coarse matching: find the nearest tree to the point $\boldsymbol{p}$
  $T_{near} \leftarrow$ None // T_{near} represents the nearest tree
  $d_{min} \leftarrow \infty$ // d_{min} represents the minimum distance
  for each tree in M **do**
    distance $\leftarrow$ COMPUTE_DISTANCE_TO_TREE($\boldsymbol{p}$, tree)
    if distance $< d_{min}$ **then**
      $T_{near} \leftarrow$ tree
      $d_{min} \leftarrow$ distance
    end
  end
  // Fine matching: find the cylinder that point $\boldsymbol{p}$ belongs to in the nearest tree
  $C \leftarrow$ None
  if $d_{min} <$ threshold **then**
    $C \leftarrow$ FIND_CYLINDER_IN_TREE($\boldsymbol{p}$, T_{near})
  end
  return C
end

IV. Tightly Coupled LiDAR–Inertial Estimator

A. State Definition

The IESKF is a Kalman filter designed for nonlinear systems, offering a balanced tradeoff between computational efficiency and accuracy. It has been widely validated in practical applications and has performed well in various complex environments [26], [31]. Therefore, we employ the IESKF framework to fuse point-to-plane and point-to-cylinder observations. The detailed algorithmic procedure of IESKF can be found in [26] and [32]. The state vector is defined as

$$\boldsymbol{x} = \begin{bmatrix} \left(\mathbf{R}_b^w\right)^T & (\boldsymbol{p}^w)^T & (\boldsymbol{v}^w)^T & \boldsymbol{b}_a^T & \boldsymbol{b}_g^T \end{bmatrix}^{\mathrm{T}} \in \mathrm{SO}(3) \times \mathbb{R}^{12} \tag{6}$$

where $\mathbf{R}_b^w$ represents the rotation matrix of the IMU frame (b-frame) relative to the world frame (w-frame); $\boldsymbol{p}^w$ and $\boldsymbol{v}^w$ represent the translation and velocity vector of the b-frame relative to the w-frame, respectively; and $\boldsymbol{b}_a$ and $\boldsymbol{b}_g$ represents the accelerometer and gyroscope biases, respectively. The estimated state and error state are defined as

$$\begin{aligned} \hat{\boldsymbol{x}} &= \begin{bmatrix} \left(\hat{\boldsymbol{R}}_b^w\right)^T & \left(\hat{\boldsymbol{p}}^w\right)^T & \left(\hat{\boldsymbol{v}}^w\right)^T & \hat{\boldsymbol{b}}_a^T & \hat{\boldsymbol{b}}_g^T \end{bmatrix}^{\mathrm{T}} \in \mathrm{SO}(3) \times \mathbb{R}^{12} \\ \delta\boldsymbol{x} &= \boldsymbol{x} \boxminus \hat{\boldsymbol{x}} = \begin{bmatrix} \delta\boldsymbol{\theta}^T & \delta\boldsymbol{p}^T & \delta\boldsymbol{v}^T & \delta\boldsymbol{b}_a^T & \delta\boldsymbol{b}_g^T \end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^{15} \end{aligned} \tag{7}$$

where $\boxminus$ denotes the subtraction operator on manifolds, with its inverse operation $\boxplus$ representing the addition operator on manifolds. The definitions and properties of these operators can be found in [26].

B. State Propagation

Before the observation update, the error covariance matrix is propagated using the motion model. While our system shares core motion modeling principles with FAST-LIO2 [26], we implement a distinct approach to IMU bias modeling. Specifically, we adopt a first-order Gauss–Markov process [22], [33], [34] to model IMU bias, which captures both short- and long-term error characteristics. This differs from FAST-LIO2's random walk model, which primarily accounts for short-term bias variations. The discrete IMU bias model employed is defined as

$$\begin{aligned} \delta\boldsymbol{b}_{a,k+1} &= \left(1 - \frac{1}{T_a}\Delta t\right)\delta\boldsymbol{b}_{a,k} + \boldsymbol{n}_a \\ \delta\boldsymbol{b}_{g,k+1} &= \left(1 - \frac{1}{T_g}\Delta t\right)\delta\boldsymbol{b}_{g,k} + \boldsymbol{n}_g \end{aligned} \tag{8}$$

where T_a and T_g represent the correlation time of the accelerometer bias and gyroscope bias, respectively, and $\boldsymbol{n}_a$ and $\boldsymbol{n}_g$ represent the Gaussian white noise of the accelerometer bias and gyroscope bias, respectively.

The other error-state transition equations of $\delta\boldsymbol{x} = \boldsymbol{x} \boxminus \hat{\boldsymbol{x}}$ are as follows:

$$\begin{aligned} \delta\boldsymbol{\theta}_{k+1} &= \mathrm{Exp}\left(-\left(\boldsymbol{\omega}^b - \boldsymbol{b}_g\right)\Delta t\right)\delta\boldsymbol{\theta}_k - \delta\boldsymbol{b}_g\Delta t - \boldsymbol{n}_\theta \\ \delta\boldsymbol{p}_{k+1} &= \delta\boldsymbol{p}_k + \delta\boldsymbol{v}\Delta t \\ \delta\boldsymbol{v}_{k+1} &= \delta\boldsymbol{v}_k - \hat{\mathbf{R}}_b^w\Delta t\delta\boldsymbol{b}_a - \hat{\mathbf{R}}_b^w\left(\left(\boldsymbol{f}^b - \boldsymbol{b}_a\right)\times\right)\Delta t\delta\boldsymbol{\theta}_k \\ &\quad - \boldsymbol{n}_v \end{aligned} \tag{9}$$

where Exp represents the exponential mapping from Lie algebra to Lie group; $\boldsymbol{n}_\theta$ and $\boldsymbol{n}_v$ represent the Gaussian white noise of the attitude and velocity, respectively; and $(\cdot\times)$ represents the conversion of a vector to a skew-symmetric matrix.

C. Point-to-Cylinder Measurement Model

The point observed by the LiDAR is represented as $\boldsymbol{p}_i^l$, where i represents the ith point and l represents the LiDAR frame (l-frame). Assuming that the LiDAR-IMU extrinsic parameters have been calibrated, the ith point $\boldsymbol{p}_i^b$ in b-frame can be obtained by transforming the point $\boldsymbol{p}_i^l$, i.e., $\boldsymbol{p}_i^b = \mathbf{R}_l^b(\boldsymbol{p}_i^l - \boldsymbol{p}_l^b)$. Here, $\boldsymbol{p}_l^b$ represents the translation vector of the l-frame relative to the b-frame, and $\mathbf{R}_l^b$ represents the rotation matrix of the l-frame relative to the b-frame.

Given the point-cloud set $P_c = \{\boldsymbol{p}_i^b\}$ of the pole-like object obtained from semantic segmentation, where i represents the ith point, the cylinder $\boldsymbol{c} = (\boldsymbol{u}^T, \boldsymbol{q}^T, r)^T$ to which the point $\boldsymbol{p}_i^b$ belongs can be determined using the data association method described in Section III-B. The distance from the point to the cylinder surface is defined as

$$d_{\boldsymbol{c},\boldsymbol{p}_i^b} = \left\|(\boldsymbol{u}\times)\left(\mathbf{R}_b^w\boldsymbol{p}_i^b + \boldsymbol{p}^w - \boldsymbol{q}\right)\right\|_2 - r. \tag{10}$$

The residual of $\boldsymbol{p}_i^b$ to the cylinder can be written as

$$\begin{aligned} h_c\left(\boldsymbol{x}_k, \boldsymbol{p}_i^b\right) &= d_{\boldsymbol{c},\boldsymbol{p}_i^b} \\ &= h_c\left(\hat{\boldsymbol{x}}_k \boxplus \delta\boldsymbol{x}_k, \boldsymbol{p}_i^b\right) + \boldsymbol{n}_i \\ &\approx h_c\left(\hat{\boldsymbol{x}}_k, \boldsymbol{p}_i^b\right) + \mathbf{H}_{c,i}\left(\hat{\boldsymbol{x}}_k, \boldsymbol{p}_i^b\right)\delta\boldsymbol{x}_k + \boldsymbol{n}_i \end{aligned} \tag{11}$$

where $\boldsymbol{n}_i$ represents the Gaussian white noise and $\mathbf{H}_{c,i}$ represents the Jacobian matrix of the point-to-cylinder observation equation, which is defined as

$$\mathbf{H}_{c,i}\left(\hat{\boldsymbol{x}}_k, \boldsymbol{p}_i^b\right) = \left.\frac{\partial h_c\left(\hat{\boldsymbol{x}}_k \boxplus \delta\boldsymbol{x}_k, \boldsymbol{p}_i^b\right)}{\partial\delta\boldsymbol{x}_k}\right|_{\delta\boldsymbol{x}_k=0}$$

$$= \begin{bmatrix} \mathbf{H}_{\delta\boldsymbol{\theta}} & \mathbf{H}_{\delta\boldsymbol{p}} & \mathbf{0}_{1\times 9} \end{bmatrix} \tag{12}$$

where $\mathbf{H}_{\delta\boldsymbol{\theta}}$ and $\mathbf{H}_{\delta\boldsymbol{p}}$ represent the Jacobian matrices of the residual with respect to the attitude error vector and position error vector, respectively, as follows:

$$\begin{aligned} \mathbf{H}_{\delta\boldsymbol{\theta}} &= -\frac{\left((\boldsymbol{u}\times)\left(\mathbf{R}_b^w \boldsymbol{p}_i^b + \boldsymbol{p}^w - \boldsymbol{q}\right)\right)^T}{\left\|(\boldsymbol{u}\times)\left(\mathbf{R}_b^w \boldsymbol{p}_i^b + \boldsymbol{p}^w - \boldsymbol{q}\right)\right\|_2}(\boldsymbol{u}\times)\mathbf{R}_b^w\left(\boldsymbol{p}_i^b\times\right) \\ \mathbf{H}_{\delta\boldsymbol{p}} &= \frac{\left((\boldsymbol{u}\times)\left(\mathbf{R}_b^w \boldsymbol{p}_i^b + \boldsymbol{p}^w - \boldsymbol{q}\right)\right)^T}{\left\|(\boldsymbol{u}\times)\left(\mathbf{R}_b^w \boldsymbol{p}_i^b + \boldsymbol{p}^w - \boldsymbol{q}\right)\right\|_2}(\boldsymbol{u}\times). \end{aligned} \tag{13}$$

D. Point-to-Plane Measurement Model

The residual of the point $\boldsymbol{p}_i^b$, which is labeled as building or ground, to the plane can be written as

$$\begin{aligned} h_\pi\left(\boldsymbol{x}_k, \boldsymbol{p}_i^b\right) &= h_\pi\left(\hat{\boldsymbol{x}}_k \boxplus \delta\boldsymbol{x}_k, \boldsymbol{p}_i^b\right) + \boldsymbol{n}_i \\ &\approx h_\pi\left(\hat{\boldsymbol{x}}_k, \boldsymbol{p}_i^b\right) + \mathbf{H}_{\pi,i}\left(\hat{\boldsymbol{x}}_k, \boldsymbol{p}_i^b\right)\delta\boldsymbol{x}_k + \boldsymbol{n}_i \end{aligned} \tag{14}$$

where $\boldsymbol{n}_i$ represents the Gaussian white noise and $\mathbf{H}_{\pi,i}$ represents the Jacobian matrix of the point-to-plane observation equation, which is defined as

$$\begin{aligned} \mathbf{H}_{\pi,i}\left(\hat{\boldsymbol{x}}_k, \boldsymbol{p}_i^b\right) &= \left.\frac{\partial h_\pi\left(\hat{\boldsymbol{x}}_k \boxplus \delta\boldsymbol{x}_k, \boldsymbol{p}_i^b\right)}{\partial\delta\boldsymbol{x}_k}\right|_{\delta\boldsymbol{x}_k=0} \\ &= \begin{bmatrix} -\boldsymbol{u}^T\mathbf{R}_b^w\left(\boldsymbol{p}_i^b\times\right) & \boldsymbol{u}^T & \mathbf{0}_{1\times 9} \end{bmatrix} \end{aligned} \tag{15}$$

where $\boldsymbol{u}$ represents the normal vector of the plane.

It should be noted that the results of semantic segmentation are not absolutely accurate. Therefore, innovative robust filtering is utilized in the state estimation process to filter out the incorrect segmentation results.

E. Maximum A Posteriori Estimation

Finally, the objective function of the state estimation problem can be obtained by combining the prior information and the point-to-plane and point-to-cylinder residuals

$$\begin{aligned} \min_{\delta\boldsymbol{x}_k}\Bigg(&\left\|\boldsymbol{x}_k \boxminus \hat{\boldsymbol{x}}_k\right\|_{\hat{\mathbf{P}}_k}^2 + \sum_{i=1}^{N_\pi}\left\|h_\pi\left(\hat{\boldsymbol{x}}_k, \boldsymbol{p}_i^b\right) + \mathbf{H}_{\pi,i}\left(\hat{\boldsymbol{x}}_k, \boldsymbol{p}_i^b\right)\delta\boldsymbol{x}_k\right\|_{\boldsymbol{\Sigma}_\pi}^2 \\ &+ \sum_{i=1}^{N_c}\left\|h_c\left(\hat{\boldsymbol{x}}_k, \boldsymbol{p}_i^b\right) + \mathbf{H}_{c,i}\left(\hat{\boldsymbol{x}}_k, \boldsymbol{p}_i^b\right)\delta\boldsymbol{x}_k\right\|_{\boldsymbol{\Sigma}_c}^2\Bigg) \end{aligned} \tag{16}$$

where N_π and N_c represent the number of point-to-plane and point-to-cylinder observations, respectively; and $\hat{\mathbf{P}}_k$, $\boldsymbol{\Sigma}_\pi$, and $\boldsymbol{\Sigma}_c$ represent the observation noise covariance matrices of the prior information, the point-to-plane, and point-to-cylinder observations, respectively. The setting method of $\boldsymbol{\Sigma}_\pi$ is the same as that in FF-LINS [35], while we use the residual covariance matrix of the cylinder fitting as $\boldsymbol{\Sigma}_c$. For a series of point clouds $\{\boldsymbol{p}_i\}$, $i = 1, 2, \ldots, N$ belonging to the cylinder $(\boldsymbol{c}, \boldsymbol{q}, r)$, first calculate the average distance d from each point to the cylinder surface

$$d = \frac{1}{N}\sum_{i=1}^{N}\left\|(\boldsymbol{u}\times)\left(\boldsymbol{p}_i - \boldsymbol{q}\right)\right\|_2 - r. \tag{17}$$

Then, the observation variance of the cylinder can be obtained by calculating

$$\sigma_c^2 = \frac{1}{N}\sum_{i=1}^{N}\left(\left\|(\boldsymbol{u}\times)\left(\boldsymbol{p}_i - \boldsymbol{q}\right)\right\|_2 - r - d\right)^2. \tag{18}$$

Hence, the covariance matrix of the cylinder is obtained

$$\boldsymbol{\Sigma}_c = \left[\sigma_c^2\right]. \tag{19}$$

Through this approach, we achieve adaptive weight adjustment of point-to-cylinder measurement.

V. Experiments and Results

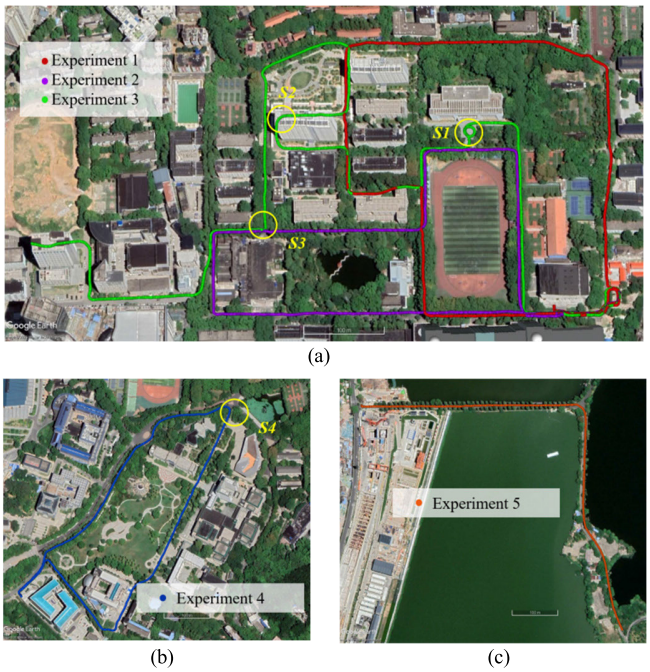
A. Implementation and Evaluation Setup

The proposed SE-LIO is implemented in C++ based on the robot operating system (ROS). The semantic segmentation model is retrained on data collected at Wuhan University. The proposed SE-LIO employs an adaptive piecewise fitting cylinder model, necessitating the configuration of the max fitting depth. This section sets the max fitting depth to 3, with a comprehensive explanation to follow in Section IV-D. It is important to note that the results are deterministic in each run. The system was run in real time on a desktop PC (Intel Core i7-11700 CPU @ 2.50 GHz, 32-GB RAM, and an NVIDIA GTX 1650 GPU) under the ROS framework. It should be noted that different data used the same parameters for the same tested method.

The employed public dataset is the MCD dataset [36], which is collected over large-scale campus areas. The dataset includes 10-Hz Livox Mid70 LiDAR and 400-Hz VN200 IMU data. We use six sequences from the dataset, including *k*th_day_06, *k*th_day_09, *k*th_day_10, *k*th_night_01, *k*th_night_04, and *k*th_night_05. These sequences are large in scale and have a long duration, which can fully evaluate the performance of the proposed method. Each sequence has a corresponding reference trajectory, and the absolute translation error (ATE) is used to evaluate the positioning performance.

The private datasets are collected with a low-speed wheeled robot with an average speed of around 1.5 m/s. The robot uses a solid-state LiDAR with a frame rate of 10 Hz (Livox Mid-70), an industrial-grade MEMS IMU (ADI ADIS16465 with a gyroscope bias instability of 2°/h and a frame rate of 200 Hz), and a dual-antenna GNSS receiver with a frame rate of 1 Hz (NovAtel OEM-718D). The GNSS real-time kinematic (RTK) technique is adopted to achieve high-accuracy positioning. All sensors are synchronized through hardware triggers to the GNSS time. The ground-truth system is a high-accuracy GNSS/INS integrated navigation system using the GNSS-RTK and a navigation-grade IMU. The ground truth (with an accuracy of 0.02 m for position and 0.01° for attitude) is generated by postprocessing software. Therefore, the positioning performance was evaluated based on absolute and relative pose errors. The absolute pose error is composed of ATE and absolute rotation error (ARE), and the relative pose error is composed of relative translation error (RTE) and relative rotation error (RRE).

The test environments on the private dataset include campus and park areas, as shown in Fig. 7. The experiments, numbered



(a) (b) (c)

Fig. 7. Test environments on private dataset. S1–S3 represent the degraded scenes in Experiment 2, and S4 represents the degraded scene in Experiment 4. (a) Experiments 1–3. (b) Experiment 4. (c) Experiment 5.

1–4, as shown in Fig. 7(a) and (b), were conducted at Wuhan University, where the environment featured numerous trees and buildings. The last experiment, as shown in Fig. 7(c), was conducted in Donghu, Wuhan, a large lake with artificial roads and trees on both sides. All the test environments contain many dynamic objects. S1∼S3 in Fig. 7 represent the degraded scenes of Experiment 2, and S4 represents the degraded scene of Experiment 4. The degradation of these scenes is due to the lack of plane features in these scenes.

To underscore the merits of the proposed SE-LIO, we conducted a comparative analysis with the LIO system that solely utilizes plane features. The proposed method is referred to as SE-LIO. The baseline method is referred to as Ori-LIO, which only uses plane features. The performance of Ori-LIO is comparable to that of FAST-LIO2 [26], but it has been further optimized, as mentioned in Section IV-B. We also tested the performances of state-of-the-art methods, including FAST-LIO2, FF-LINS [35], FASTER-LIO [31], and PV-LIO [37], on both the public and private datasets. To verify the effectiveness of the cylindrical features, we also conducted ablation experiments, i.e., testing the positioning accuracy that solely removed the unstructured point clouds and compared it with the proposed SE-LIO. The method that solely removes unstructured point clouds is referred to as SE-LIO-RU, where RU stands for removing unstructured point clouds.

It should be noted that the semantic segmentation model is trained on data collected at Wuhan University, and the segmentation effect of unstructured point clouds (leaves and dynamic objects) on public datasets is not good. In addition, we do not compare it with SLOAM because SLOAM is specifically designed for spinning LiDAR. Consequently, its semantic segmentation and cylinder-fitting methods are incompatible with our dataset.

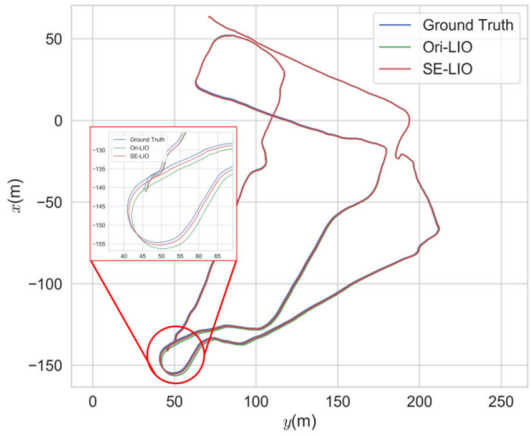


Fig. 8. Trajectories of the proposed SE-LIO and Ori-LIO on sequence *k*th_night_04.

B. Evaluation of the Positioning Accuracy

1) Public MCD Dataset: We obtained the ATE results for the public MCD dataset, as shown in Table I. The superior results are highlighted in bold. From Table I, it is evident that the baseline algorithm Ori-LIO outperforms FAST-LIO2, FF-LINS, FASTER-LIO, and PV-LIO in most sequences. However, SE-LIO-RU performs worse than Ori-LIO in some sequences. This is attributed to the semantic segmentation model used in these experiments. The model was trained on the RobNav dataset, leading to suboptimal semantic segmentation results on the MCD dataset and a subsequent decrease in positioning accuracy. Nevertheless, in most sequences, the proposed SE-LIO achieves the best results, with an average ATE accuracy improvement of 17.1% compared to Ori-LIO. This indicates that the proposed SE-LIO outperforms other methods in positioning performance on the public dataset. Fig. 8 shows the positioning trajectories of the proposed SE-LIO and Ori-LIO on sequence *k*th_night_04. It can be seen that the positioning trajectory of the proposed SE-LIO is closer to the reference trajectory.

2) Private Dataset: The ARE and ATE are shown in Table II, with the superior result among the five emphasized in bold, and E1∼E5 represent Experiments 1∼5, respectively. As depicted in Table II, Ori-LIO demonstrates superior performance compared to FAST-LIO2, FF-LINS, FASTER-LIO, and PV-LIO across multiple test scenarios. SE-LIO-RU further enhances positioning accuracy by effectively removing unstructured point clouds, reducing the ATE by 19.6% compared to Ori-LIO. This improvement is particularly evident in tree-rich environments where traditional methods struggle with noise from unstructured point clouds. SE-LIO builds upon these advantages and incorporates additional cylindrical features, achieving even higher ATE accuracy with a 43.1% improvement compared to Ori-LIO. Specifically, in Experiment 5, the test scene is situated by a lake. During the majority of Experiment 5, the point cloud in the horizontal direction consists only of unstructured point clouds, such as trees and dynamic objects, lacking structured point clouds like buildings. Therefore, the absolute positioning accuracy is greatly improved after removing unstructured point clouds.

TABLE I
ATEs ON THE MCD DATASET

ATE(m)	FF-LINS	FAST-LIO2	FASTER-LIO	PV-LIO	Ori-LIO	SE-LIO-RU	SE-LIO
kth_day_06	2.79	0.81	**0.58**	1.43	0.72	0.61	**0.58**
kth_day_09	0.94	0.41	0.56	1.28	0.24	0.43	**0.24**
kth_day_10	0.68	**0.58**	1.00	1.30	0.80	1.39	0.67
kth_night_01	2.14	1.15	**0.71**	1.72	0.90	0.91	0.82
kth_night_04	0.88	0.62	0.45	1.14	0.46	0.30	**0.25**
kth_night_05	1.06	0.67	1.74	1.17	0.57	1.11	**0.50**
MEAN	1.41	0.71	0.84	1.34	0.61	0.79	**0.51**

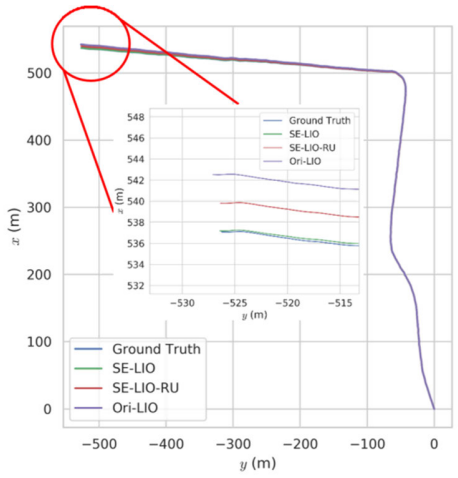


Fig. 9. Trajectories of the three methods on Experiment 5.

Furthermore, the proposed method utilizes pole-like objects like tree trunks for positioning, thereby strengthening the horizontal constraint and enhancing the absolute positioning accuracy compared to SE-LIO-RU.

Fig. 9 shows the trajectories estimated by Ori-LIO, SE-LIO-RU, and SE-LIO in Experiment 5. SE-LIO exhibits the least drift, aligning more closely with the ground truth. In contrast, both SE-LIO-RU and Ori-LIO demonstrate larger drift. The RTE and RRE were also evaluated to provide insight into short-term accuracy. The results are presented in Table III, with the superior result among the three emphasized in bold. As exhibited in Table III, the proposed SE-LIO outperforms the other two methods in most tests. The short-term accuracy of the proposed SE-LIO is improved, reflecting the system's superior robustness.

The mapping result of SE-LIO on public and private datasets is shown in Fig. 10. In Fig. 10(a), the trajectory repeatedly passes through the same position, and the point-cloud map does not overlap, indicating that SE-LIO can maintain consistency in the mapping process and has good positioning accuracy. In Fig. 10(b), SE-LIO can accurately fit the cylinder even in an environment with few pole-like objects, indicating that SE-LIO has good cylinder-fitting capabilities. In Fig. 10(c) and (d), SE-LIO successfully fits the cylinder for the curved and straight tree point clouds on both sides of the road, indicating that SE-LIO can adapt well to environments with many curved and straight pole-like objects.

C. Evaluation of the Robustness

We selected several representative scenes on the private dataset to evaluate the robustness of the proposed method in tree-rich environments, as depicted in Fig. 7. Scenes S1–S3 represent the degraded conditions of Experiment 2, while Scene S4 corresponds to the degraded condition of Experiment 4. The robot performed a turning maneuver in all scenes. Pole-like object features are lacking in Scene S2 but are observable in Scenes S1, S3, and S4. The corresponding scenes of S1∼S4 are shown in Fig. 11.

The fit cylinders and the robot trajectory in scenes selected for evaluating the robustness of the proposed method are shown in Fig. 12. It can be seen that a large number of cylinders can be fit in S1, S3, and S4, while almost no cylinders are fit in S2 due to the lack of pole-like objects.

The RTEs curve of the robot over a distance of 25 m in these scenes is depicted in Fig. 13. In each subfigure, the upper part represents the RTEs of the robot, and the lower part represents the number of cylindrical features observed by the robot. As inferred from Fig. 13, in Scenes S1, S3, and S4, SE-LIO demonstrates superior positioning accuracy compared to Ori-LIO and SE-LIO-RU. It suggests that the proposed method exhibits robustness in environments rich in trees. The cylindrical features significantly enhance the robot's positioning accuracy during turning maneuvers by providing additional horizontal constraint information, compensating for the limitations of plane features during such maneuvers. Notably, in Scene S2, the positioning accuracy of SE-LIO is comparable to that of Ori-LIO and SE-LIO-RU, indicating that the proposed method does not compromise accuracy in scenes devoid of pole-like object features.

In Scene S4, the positioning accuracy of SE-LIO-RU is notably the least optimal. It can be attributed to the fact that in Scene S4, the LiDAR scans fewer plane features, such as buildings. Ori-LIO makes use of unstructured point clouds while SE-LIO-RU removes them. Despite the unreliability of the plane features extracted from unstructured point clouds, they provide horizontal constraints, resulting in better positioning accuracy than SE-LIO-RU. However, SE-LIO, which is based on SE-LIO-RU, utilizes cylindrical features that provide

TABLE II
AREs AND ATEs ON THE PRIVATE DATASET

ARE/ ATE (deg / m)	FF-LINS	FAST-LIO2	FASTER-LIO	PV-LIO	Ori-LIO	SE-LIO-RU	SE-LIO
E1	**0.42 / 1.62**	2.46 / 5.61	2.92 / 3.86	2.76 / 3.36	0.86 / 2.59	0.68 / 2.34	0.55/ 1.85
E2	0.37 / 1.38	3.30 / 6.54	2.52 / 0.88	1.85 / 1.56	0.62/ 1.25	**0.28** / 0.89	0.34 / **0.75**
E3	0.48 / 1.29	1.42 / 2.00	2.22 / 3.34	2.48 / 3.27	0.53/ 1.25	0.52/ 1.32	**0.41 / 0.94**
E4	0.89 / 4.04	2.49 / 5.16	3.15 / 5.24	2.73 / 4.43	0.56 / 1.21	**0.25** / 0.92	0.29/ **0.63**
E5	0.89 / 3.02	2.47 / 1.47	2.13 / 1.14	4.22 / 1.77	0.33/ 1.37	0.21 / 0.67	**0.20 / 0.20**
MEAN	0.61 / 2.27	2.43 / 4.16	2.58 / 2.89	2.80 / 2.88	0.58 / 1.53	0.39/ 1.23	**0.36/ 0.87**

TABLE III
RREs AND RTEs ON THE PRIVATE DATASET

RRE / RTE (deg / %)	25m			100m			200m		
	Ori-LIO	SE-LIO-RU	SE-LIO	Ori-LIO	SE-LIO-RU	SE-LIO	Ori-LIO	SE-LIO-RU	SE-LIO
E1	0.31 / **0.86**	0.30 / 0.89	**0.30** / 0.87	0.46 / 0.62	0.37 / 0.61	**0.34 / 0.61**	0.58 / 0.65	0.48 / 0.59	**0.40 / 0.58**
E2	0.15 / 0.87	0.10 / 0.87	**0.10 / 0.82**	0.30 / **0.61**	0.17 / 0.64	**0.15** / 0.63	0.35 / **0.43**	0.24 / 0.45	**0.20** / 0.47
E3	0.20 / 1.01	0.16 / 0.98	**0.15 / 0.92**	0.32 / 0.72	0.29 / 0.70	**0.24 / 0.67**	0.40 / 0.56	0.40 / 0.55	**0.31 / 0.52**
E4	0.16 / 0.84	0.13 / 0.85	**0.12 / 0.77**	0.27 / 0.61	0.17 / 0.57	**0.16 / 0.53**	0.35 / 0.50	**0.21** / 0.42	0.22 / **0.38**
E5	0.10 / 0.48	0.10 / 0.46	**0.09 / 0.43**	0.18 / 0.41	0.14 / **0.32**	**0.10** / 0.34	0.28 / 0.44	0.19 / 0.32	**0.11 / 0.31**
MEAN	0.18 / 0.81	0.16 / 0.81	**0.15 / 0.76**	0.31 / 0.59	0.23 / 0.57	**0.20 / 0.56**	0.39 / 0.52	0.30 / 0.47	**0.25 / 0.45**

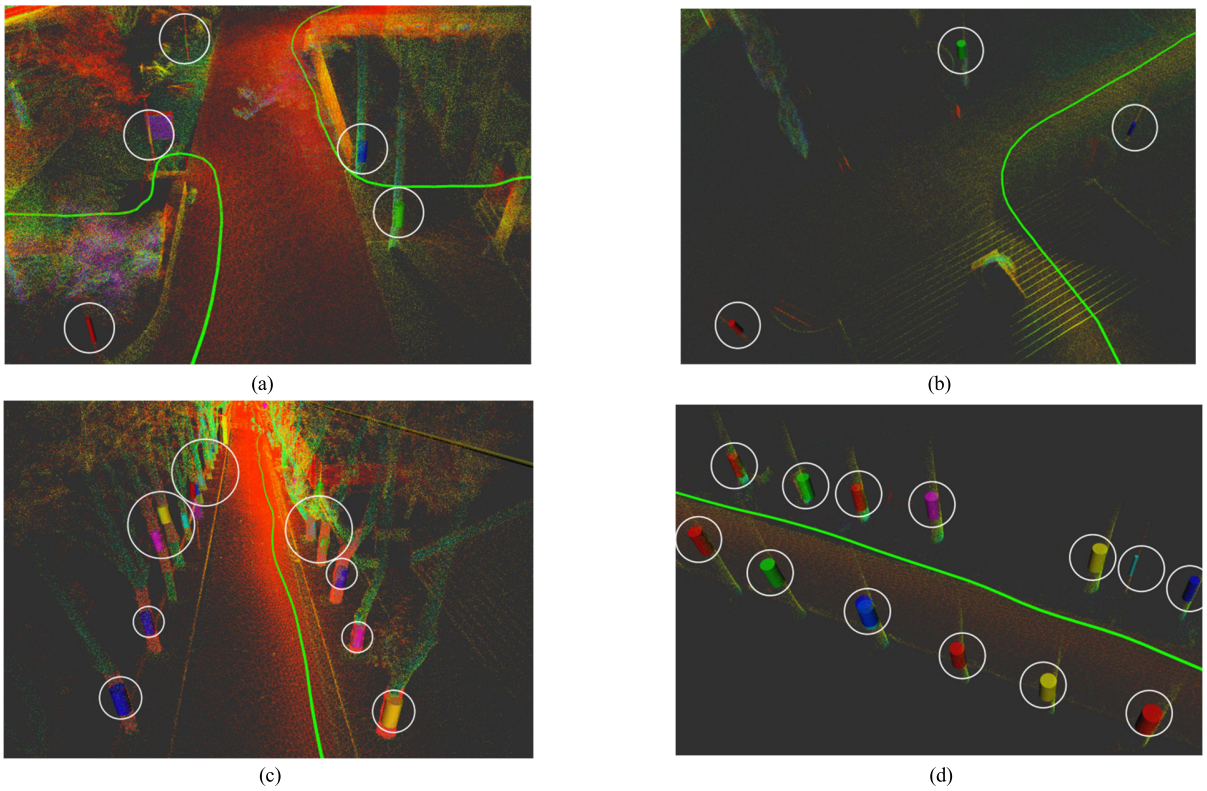


Fig. 10. Mapping result of SE-LIO on public and private datasets. The cylinders in the figure represent the cylinders fit by SE-LIO, and the fit cylinders are marked with white circles. (a) *k*th_night_04 on the public dataset. (b) *k*th_day_06 on the public dataset. (c) Experiment 3 on the private dataset. (d) Experiment 4 on the private dataset.

more reliable horizontal constraints. Consequently, SE-LIO demonstrates superior positioning accuracy compared to both Ori-LIO and SE-LIO-RU.

D. Ablation Experiment

1) Impact of Point-Cloud Integration on Semantic Segmentation Accuracy: We used multiframe merging to

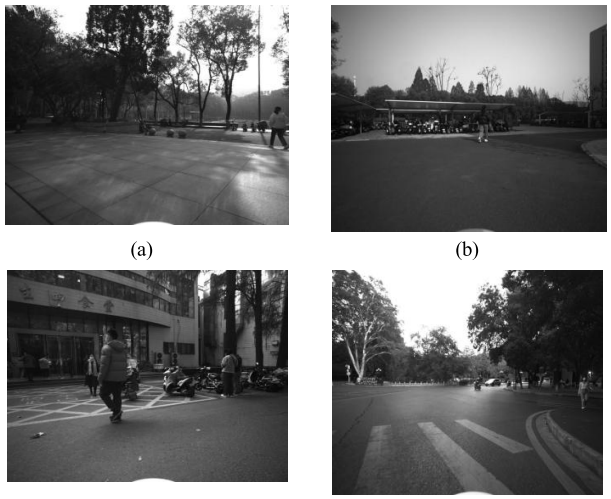


Fig. 11. Scenes selected for evaluating the robustness of the proposed method in tree-rich environments. (a) Scene S1. (b) Scene S2. (c) Scene S3. (d) Scene S4.

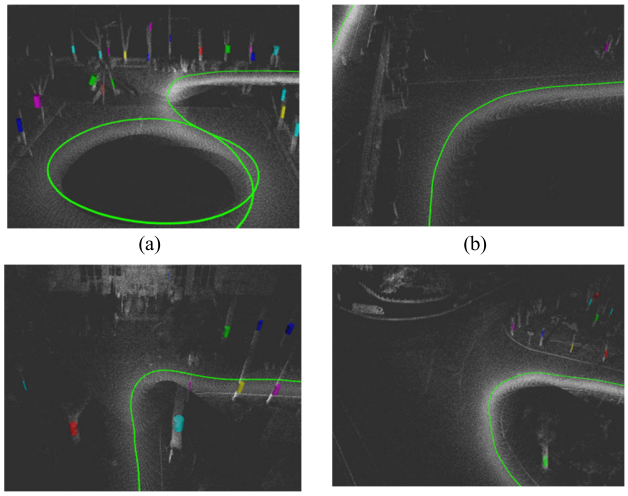


Fig. 12. Fit cylinders and the robot trajectory in scenes selected for evaluating the robustness of the proposed method. The green line represents the robot trajectory, and the cylinders are the cylinders fit by SE-LIO. The white point cloud represents the LiDAR point cloud. (a) Scene S1. (b) Scene S2. (c) Scene S3. (d) Scene S4.

achieve point-cloud integration. The LiDAR point cloud was motion-compensated using the INS pose during multiframe merging. The duration of a single frame of point cloud was 0.1 s, and the number of merged frames ranged from 1 to 6, corresponding to a merging time of 0.1–0.6 s. The test results are shown in Table IV. Here, f represents frame(s), i.e., the number of frames.

As shown in Table IV, merging consecutive point-cloud frames improves semantic segmentation accuracy by increasing point-cloud coverage. The results demonstrate that denser point clouds lead to better segmentation performance, with average intersection over union (IoU) increasing from 52% to 70% when merging up to three frames. However, the accuracy improvement plateaus beyond three merged frames. While increasing the number of merged frames reduces semantic segmentation computational frequency, it also introduces larger INS motion compensation errors. Based on these tradeoffs between segmentation accuracy, processing frequency, and motion compensation accuracy, we selected five merged frames as the optimal configuration for subsequent experiments.

TABLE IV
SEMANTIC SEGMENTATION ACCURACY UNDER DIFFERENT MERGING FRAMES

Class IoUs	Number of frames					
	1	2	3	4	5	6
car	0.51	0.76	0.77	0.77	0.75	**0.78**
motorcycle	0.11	0.44	0.56	0.54	0.55	**0.61**
road	0.93	0.96	**0.97**	**0.97**	**0.97**	**0.97**
building	0.23	0.39	0.40	0.41	0.40	**0.43**
vegetation	0.81	0.85	0.85	0.85	0.85	**0.86**
trunk	0.51	0.62	0.63	0.63	0.64	**0.67**
MEAN	0.52	0.67	0.70	0.70	0.69	**0.72**

TABLE V
ARE AND ATE OF THE METHODS WITH DIFFERENT MAX DEPTHS

ARE / ATE (deg / m)	D1	D2	D3	D4
E1	0.58 / 1.91	0.58 / 1.88	**0.55 / 1.85**	0.65 / 2.21
E2	0.34 / 0.79	**0.33 / 0.74**	0.34 / 0.75	0.34 / 0.76
E3	0.48 / 1.17	0.46 / 1.09	**0.41 / 0.94**	0.45 / 1.04
E4	0.27 / 0.64	**0.27 / 0.57**	0.29 / 0.63	0.28 / 0.67
E5	0.21 / **0.19**	0.20 / 0.20	**0.20** / 0.20	0.20 / 0.20
MEAN	0.38 / 0.94	0.37 / 0.89	**0.36 / 0.87**	0.38 / 0.97

2) Impact of Adaptive Piecewise Cylinder Fitting on Positioning Accuracy: As previously discussed, we employ a binary tree to implement adaptive piecewise fitting. Different piecewise fitting results can be obtained by adjusting the binary tree's max depth. When the max depth is set to i, the point cloud of this pole-like objects cluster will be divided into 2^{i-1} segments at most. On the private dataset, the impact of piecewise cylinder fitting on positioning accuracy was evaluated by setting the max depth of the binary tree to 1–4. A max depth of 1 indicates no piecewise fitting, and the results of different piecewise fitting depths are denoted as D1–D4.

The absolute pose error is shown in Table V, where the best result among the four is highlighted in bold. As can be seen from Table V, when the max depth of the binary tree is 1, the positioning accuracy of the three tests is suboptimal. In Experiments 1–4, positioning accuracy is improved when the max depth is either 2 or 3, compared to a max depth of 1. However, increasing the max depth to four decreases positioning accuracy. These results indicate that piecewise fitting has a certain impact on positioning accuracy, with optimal positioning accuracy achieved at a max depth of 3. Further increases in max depth do not enhance accuracy but increase computational complexity. Consequently, the max depth of the binary tree is set to 3.

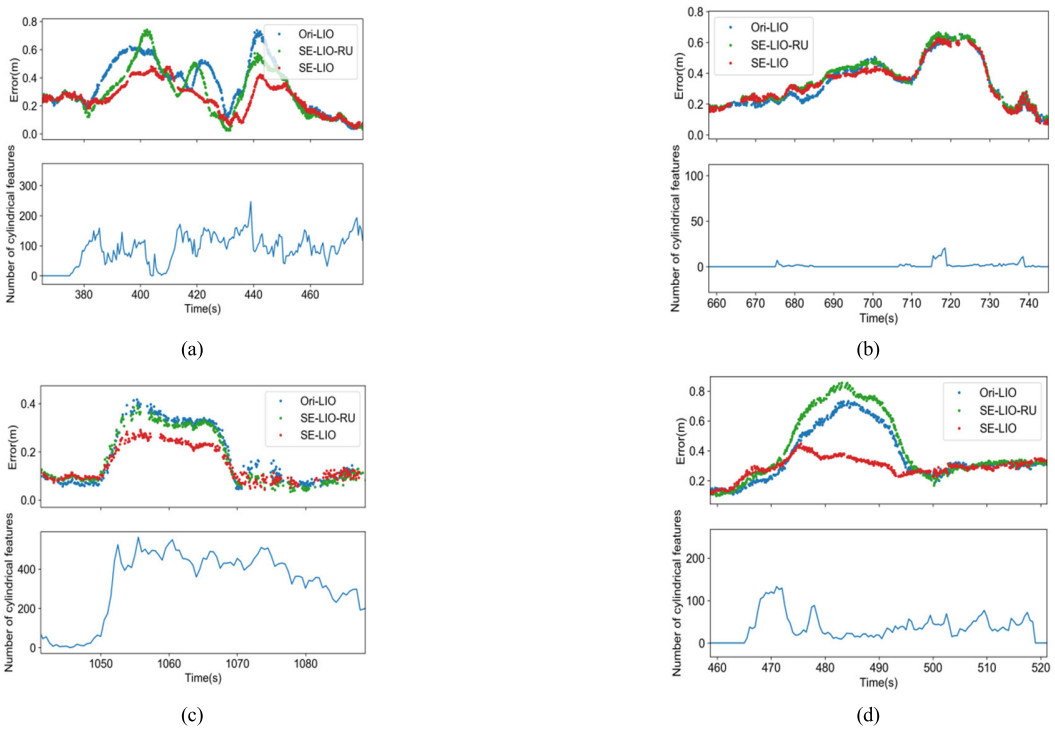


Fig. 13. RTE curve of the robot over a distance of 25 m in Scenes S1∼S4. In each subfigure, the upper part represents the RTEs of the robot, and the lower part represents the number of cylindrical features observed by the robot. (a) Scene S1. (b) Scene S2. (c) Scene S3. (d) Scene S4.

E. Runtime Analysis

We further tested the runtime of the proposed method. The test environment comprised an Intel Core i7-11700 CPU @ 2.50 GHz, 32-GB RAM, and an NVIDIA GTX 1650 GPU. In Section IV-B, we merged five frames of point clouds (0.5 s) for semantic segmentation and LIO positioning. For a 0.5-s point cloud, the average runtime for semantic segmentation was 392.8 ms, while the LIO fusion took 33.5 ms. Semantic segmentation accounted for approximately 90% of the total runtime, as we did not quantize or prune the model, resulting in excessive computational overhead. This indicates that our runtime efficiency can be further optimized in future work.

VI. Conclusion

In this study, we propose a semantic-enhanced solid-state-LIO method for environments abundant with trees, such as campuses and parks. The method integrates and compensates multiple LiDAR frames using the INS pose to address the problem of low semantic segmentation accuracy due to sparse point clouds. Semantic information is then employed to enhance the positioning performance, including removing unstructured point clouds and constructing point-to-cylinder constraints using the cylindrical features of pole-like objects. An adaptive piecewise fitting method is proposed that the pole-like object is segmented into multiple cylinders, obtaining more accurate cylindrical features. Hence, the positioning accuracy can be improved in scenes with curved tree trunks. Experimental results demonstrate that SE-LIO outperforms the baseline method in terms of positioning accuracy and robustness.

By employing the proposed semantic-enhanced method, the positioning accuracy and robustness can be significantly improved. Nevertheless, the efficiency of the semantic segmentation model remains suboptimal. Techniques such as model pruning and quantization can be employed to enhance segmentation efficiency. Besides, incorporating additional semantic information holds the potential to further refine positioning accuracy. In addition, for spinning LiDARs with fewer laser beams, such as 8 or 16, a rotating mechanism can be employed to improve the point-cloud coverage and the segmentation accuracy. Meanwhile, the INS can be used to improve the accuracy of semantic segmentation.

References

[1] P. Biber and W. Strasser, "The normal distributions transform: A new approach to laser scan matching," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Las Vegas, NV, USA, Jun. 2003, pp. 2743–2748, doi: 10.1109/IROS.2003.1249285.

[2] A. Thakur and P. Rajalakshmi, "LiDAR-based optimized normal distribution transform localization on 3-D map for autonomous navigation," *IEEE Open J. Instrum. Meas.*, vol. 3, 2024, Art. no. 8500211, doi: 10.1109/OJIM.2024.3412219.

[3] M. Magnusson, A. Lilienthal, and T. Duckett, "Scan registration for autonomous mining vehicles using 3D-NDT," *J. Field Robot.*, vol. 24, no. 10, pp. 803–827, 2007, doi: 10.1002/rob.20204.

[4] A. Das, J. Servos, and S. L. Waslander, "3D scan registration using the normal distributions transform with ground segmentation and point cloud clustering," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2013, pp. 2207–2212, doi: 10.1109/ICRA.2013.6630874.

[5] C. Yuan, W. Xu, X. Liu, X. Hong, and F. Zhang, "Efficient and probabilistic adaptive voxel mapping for accurate online LiDAR odometry," *IEEE Robot. Autom. Lett.*, vol. 7, no. 3, pp. 8518–8525, Jul. 2022.

[6] 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. (IROS)*, Oct. 2018, pp. 4758–4765, doi: 10.1109/IROS.2018.8594299.

[7] I. Bogoslavskyi and C. Stachniss, "Fast range image-based segmentation of sparse 3D laser scans for online operation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2016, pp. 163–169, doi: 10.1109/IROS.2016.7759050.

[8] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep learning for 3D point clouds: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 12, pp. 4338–4364, Dec. 2021, doi: 10.1109/TPAMI.2020.3005434.

[9] B. Wu, A. Wan, X. Yue, and K. Keutzer, "SqueezeSeg: Convolutional neural nets with recurrent CRF for real-time road-object segmentation from 3D LiDAR point cloud," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 1887–1893, doi: 10.1109/ICRA.2018.8462926.

[10] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, "RangeNet ++: Fast and accurate LiDAR semantic segmentation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 4213–4220, doi: 10.1109/IROS40897.2019.8967762.

[11] R. A. Rosu, P. Schütt, J. Quenzel, and S. Behnke, "LatticeNet: Fast spatio-temporal point cloud segmentation using permutohedral lattices," *Auto. Robots*, vol. 46, no. 1, pp. 45–60, Jan. 2022, doi: 10.1007/s10514-021-09998-1.

[12] D. Rethage, J. Wald, J. Sturm, N. Navab, and F. Tombari, "Fully-convolutional point networks for large-scale point clouds," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, in Lecture Notes in Computer Science, vol. 11208, Cham, Switzerland. Springer, 2018, pp. 625–640, doi: 10.1007/978-3-030-01225-0_37.

[13] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 652–660. Accessed: Jun. 11, 2024. [Online]. Available: https://openaccess.thecvf.com/content_cvpr_2017/html/Qi_PointNe_Deep_Learning_CVPR_2017_paper.html

[14] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1–8. Accessed: Nov. 9, 2023. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/hash/d8bf84be3800d12f74d8b05e9b89836f-Abstract.html

[15] Q. Hu et al., "RandLA-Net: Efficient semantic segmentation of large-scale point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11105–11114, doi: 10.1109/CVPR42600.2020.01112.

[16] X. Chen, A. Milioto, E. Palazzolo, P. Giguère, J. Behley, and C. Stachniss, "SuMa++: Efficient LiDAR-based semantic SLAM," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 4530–4537, doi: 10.1109/IROS40897.2019.8967704.

[17] G. Chen, B. Wang, X. Wang, H. Deng, B. Wang, and S. Zhang, "PSF-LO: Parameterized semantic features based LiDAR odometry," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Xi'an, China, May 2021, pp. 5056–5062, doi: 10.1109/ICRA48506.2021.9561554.

[18] S. W. Chen et al., "SLOAM: Semantic LiDAR odometry and mapping for forest inventory," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 612–619, Apr. 2020, doi: 10.1109/LRA.2019.2963823.

[19] L. Li et al., "SA-LOAM: Semantic-aided LiDAR SLAM with loop closure," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 7627–7634, doi: 10.1109/ICRA48506.2021.9560884.

[20] J. Lin and F. Zhang, "Loam livox: A fast, robust, high-precision LiDAR odometry and mapping package for LiDARs of small FoV," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 3126–3131, doi: 10.1109/ICRA40945.2020.9197440.

[21] K. Li, M. Li, and U. D. Hanebeck, "Towards high-performance solid-state-LiDAR-inertial odometry and mapping," *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 5167–5174, Jul. 2021, doi: 10.1109/LRA.2021.3070251.

[22] H. Tang, X. Niu, T. Zhang, L. Wang, and J. Liu, "LE-VINS: A robust solid-state-LiDAR-enhanced visual-inertial navigation system for low-speed robots," *IEEE Trans. Instrum. Meas.*, vol. 72, pp. 1–13, 2023, doi: 10.1109/TIM.2023.3260279.

[23] J. Zhang and S. Singh, "LOAM: LiDAR odometry and mapping in real-time," in *Proc. Robot., Sci. Syst. Conf.*, Jul. 2014, pp. 1–9, doi: 10.15607/RSS.2014.X.007.

[24] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "LIO-SAM: Tightly-coupled LiDAR inertial odometry via smoothing and mapping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 5135–5142, doi: 10.1109/IROS45743.2020.9341176.

[25] W. Xu and F. Zhang, "FAST-LIO: A fast, robust LiDAR-inertial odometry package by tightly-coupled iterated Kalman filter," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 3317–3324, Apr. 2021, doi: 10.1109/LRA.2021.3064227.

[26] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "FAST-LIO2: Fast direct LiDAR-inertial odometry," *IEEE Trans. Robot.*, vol. 38, no. 4, pp. 2053–2073, Aug. 2022, doi: 10.1109/TRO.2022.3141876.

[27] R. Schnabel, R. Wahl, and R. Klein, "Efficient ransac for point-cloud shape detection," *Comput. Graph. Forum*, vol. 26, no. 2, pp. 214–226, 2007, doi: 10.1111/j.1467-8659.2007.01016.x.

[28] G. Li, X. Huang, and S. Li, "A novel circular points-based self-calibration method for a camera's intrinsic parameters using RANSAC," *Meas. Sci. Technol.*, vol. 30, no. 5, Apr. 2019, Art. no. 055005, doi: 10.1088/1361-6501/ab09c0.

[29] L. Zhou, G. Huang, Y. Mao, J. Yu, S. Wang, and M. Kaess, "$\mathcal{PLC}$-LiSLAM: LiDAR SLAM with planes, lines, and cylinders," *IEEE Robot. Autom. Lett.*, vol. 7, no. 3, pp. 7163–7170, Jul. 2022, doi: 10.1109/LRA.2022.3180116.

[30] Y. Rui, Z. Zhou, X. Cai, and L. Dong, "A novel robust method for acoustic emission source location using DBSCAN principle," *Measurement*, vol. 191, Mar. 2022, Art. no. 110812, doi: 10.1016/j.measurement.2022.110812.

[31] C. Bai, T. Xiao, Y. Chen, H. Wang, F. Zhang, and X. Gao, "Faster-LIO: Lightweight tightly coupled LiDAR-inertial odometry using parallel sparse incremental voxels," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 4861–4868, Apr. 2022, doi: 10.1109/LRA.2022.3152830.

[32] D. He, W. Xu, and F. Zhang, "Kalman filters on differentiable manifolds," 2021, *arXiv:2102.03804*.

[33] E.-H. Shin, "Estimation techniques for low-cost inertial navigation," Dept. Geomatics Eng., Univ. Calgary, Calgary, AB, Canada, 2005.

[34] H. Tang, T. Zhang, X. Niu, J. Fan, and J. Liu, "Impact of the earth rotation compensation on MEMS-IMU preintegration of factor graph optimization," *IEEE Sensors J.*, vol. 22, no. 17, pp. 17194–17204, Sep. 2022, doi: 10.1109/JSEN.2022.3192552.

[35] H. Tang, T. Zhang, X. Niu, L. Wang, L. Wei, and J. Liu, "FF-LINS: A consistent frame-to-frame solid-state-LiDAR-inertial state estimator," *IEEE Robot. Autom. Lett.*, vol. 8, no. 12, pp. 8525–8532, Dec. 2023, doi: 10.1109/LRA.2023.3329625.

[36] T.-M. Nguyen et al., "MCD: Diverse large-scale multi-campus dataset for robot perception," 2024, *arXiv:2403.11496*.

[37] (Sep. 2, 2024). *KINO, HViktorTsoi/PV-LIO*. Accessed: Sep. 3, 2024. [Online]. Available: https://github.com/HViktorTsoi/PV-LIO