IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

LOG-LIO: A LiDAR-Inertial Odometry with Efficient Local Geometric Information Estimation

Kai Huang[1], Junqiao Zhao*[,2,3], Zhongyang Zhu[2,3], Chen Ye[2], Tiantian Feng[1]

***Abstract*—Local geometric information, i.e., normal and distribution of points, is crucial for LiDAR-based simultaneous localization and mapping (SLAM) because it provides constraints for data association, which further determines the direction of optimization and ultimately affects the accuracy of localization. However, estimating normal and distribution of points are time-consuming tasks even with the assistance of kdtree or volumetric maps. To achieve fast normal estimation, we look into the structure of LiDAR scan and propose a ring-based fast approximate least squares (Ring FALS) method. With the Ring structural information, estimating the normal requires only the range information of the points when a new scan arrives. To efficiently estimate the distribution of points, we extend the ikd-tree to manage the map in voxels and update the distribution of points in each voxel incrementally while maintaining its consistency with the normal estimation. We further fix the distribution after its convergence to balance the time consumption and the correctness of representation. Based on the extracted and maintained local geometric information, we devise a robust and accurate hierarchical data association scheme where point-to-surfel association is prioritized over point-to-plane. Extensive experiments on diverse public datasets demonstrate the advantages of our system compared to other state-of-the-art methods. Our open source implementation is available at https://github.com/tiev-tongji/LOG-LIO.**

***Index Terms*—Lidar-inertial odometry, SLAM, sensor fusion.**

I. INTRODUCTION

THE performance of LiDAR (-inertial)-based simultaneous localization and mapping (SLAM) system depends heavily on the registration between the LiDAR scan and the map, i.e., finding the correspondence between them based on the similarity of the local geometric information and then minimizing their distance. Local geometric information includes attributes that can represent the position, shape, and other characteristics of the local surface where a point is located.

Manuscript received: August 13, 2023; Revised October 4, 2023; Accepted October 22, 2023.

This paper was recommended for publication by Editor Javier Civera upon evaluation of the Associate Editor and Reviewers' comments. This work is supported by the National Key Research and Development Program of China (No. 2021YFB2501104). *(Corresponding Author: Junqiao Zhao.)*

[1]Kai Huang and Tiantian Feng are with the School of Surveying and Geo-Informatics, Tongji University, Shanghai, China (e-mail: huangkai@tongji.edu.cn; fengtiantian@tongji.edu.cn).

[2]Junqiao Zhao, Zhongyang Zhu and Chen Ye are with Department of Computer Science and Technology, School of Electronics and Information Engineering, Tongji University, Shanghai, China, and the MOE Key Lab of Embedded System and Service Computing, Tongji University, Shanghai, China (e-mail: zhaojunqiao@tongji.edu.cn; 2233057@tongji.edu.cn; yechen@tongji.edu.cn).

[3]Institute of Intelligent Vehicles, Tongji University, Shanghai, China.

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

The conventional method for estimating local geometric information is to evaluate the smoothness of the input scan and to locally approximate the map with geometric primitives [1]. However, accurate estimation of local geometric information requires the retrieval of neighborhood information in a dense point cloud, but for LiDAR-inertial odometry (LIO) systems, this results in a huge computational burden even with the help of kdtree or volumetric maps.

Accurate and fast estimation of local geometric information has gained increasing attention in recent studies [2]–[6]. Among them, the normal and the distribution of points are two representative attributes, since the former indicates the tangent plane of the local surface, and the latter implies the average position and shape of the point cloud sampled from the local surface. However, current LIO systems seldom incorporate the real-time estimation of the normal and the distribution of points, which hampers their pose estimation performance.

This paper presents LOG-LIO, a robust and accurate LIO system focusing on the real-time estimation of the normal of LiDAR scan points and the distribution of map points, and their rational utilization. Inspired by [7] and [8], we look into the structure of a LiDAR scan and propose a Ring-based fast approximate least squares method, namely Ring FALS. We project point cloud onto the range image to pre-build a lookup table, which represents the structural information of the specific LiDAR. With the arrival of a new scan, only the range information of the points is needed to estimate the normal. We incrementally update the distribution of points for each voxel in the map while maintaining its consistency with the normals. To balance time consumption and correctness of representation, we manage the map on the extended ikd-tree and further fix the distribution after it converges.

Similar to the FAST-LIO series [9], [10], we directly associate scan points with voxels on the map after distortion correction. For scan points that satisfy visibility and consistency checks based on normals, we devise a robust and accurate hierarchical data association scheme considering the distribution. The poses are optimized by integrating the IMU measurements as initial estimates and then using an error-state iterative Extended Kalman filter (iEKF) [9] to minimize the multi-scale point-to-surfel and point-to-plane distances.

The main contributions of this work are as follows:

- Ring FALS, modified from FALS, a normal estimator that utilizes the structural information of LiDAR scan can meet the real-time requirements of the LIO system.
- A robust and accurate hierarchical data association scheme considering the distribution of points within map

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

voxels where point-to-surfel is prioritized over point-to-plane and large-scale surfel over small-scale surfel.
- Extensive experiments on public datasets demonstrate the advantages of our LIO system compared to other state-of-the-art methods. To benefit the community, our implementation of this work is open-source at https://github.com/tiev-tongji/LOG-LIO, and we also open-source Ring FALS as an independent normal estimation tool at https://github.com/tiev-tongji/RingFalsNormal.

II. RELATED WORKS

A. Point Cloud Normal Estimation

The most commonly used method to obtain surface normals from point cloud is the least square estimation based on the neighborhood search due to its ease of implementation [11]. However, the least squares-based method is computationally expensive for LIO systems.

[7] compares the complexity of least squares approaches and proposes FALS, which simplifies the least squares loss function to completely avoid the computation of the covariance matrix for each point. [7] also reformulates the traditional least squares solution to estimate the normal by calculating the derivatives of the surface from a spherical range image (SRI). However, the number of multiplications for normals computation of SRI is greater than FALS.

Rather than least square-based solution, 3F2N [8] performs three filtering operations on the inverse depth image to estimate the normals, which has the comparative performance as FALS, but its efficiency and accuracy are strongly influenced by the filter selection.

Inspired by [7] and [8], we propose Ring FALS. We prebuild a lookup table which represent the structural information for the specific LiDAR. Compared to FALS, Ring FALS further simplifies the projection of each LiDAR scan with the assistance of ring index, while preserving accuracy.

B. Distribution of Points Estimation

The distribution of a point is represented through its 3D coordinates and the covariance matrix computed by its neighboring points. [12] proposes the generalized ICP (GICP) algorithm, which takes into account the locally planar structure of points in a probabilistic model and then minimizes the distance between distributions. But searching neighboring points to compute the covariance matrix is too time-consuming for LIO systems.

LOAM [1] does not estimate the distribution of points, but performs Eigen analysis on the associated map points to determine whether its local geometry is a line or a plane. However, the coordinates of the sparse map points cannot accurately represent the local geometric information, which leads to inaccurate constraints for the registration.

DLO [2] registers point cloud using GICP to minimize the plane-to-plane distance, which is derived from the covariance matrix of each point. It assumes that the covariance of submap can be approximated by concatenating the normals from keyframes, and the covariance of points is only computed once when the scan is acquired. However, such a normals stitching method cannot accurately reflect the local geometric information of the point cloud, which ultimately affects accuracy. LOCUS 2.0 [4] extends the work of LOCUS [5], which constructs covariance matrices for GICP-based registration based on the pre-computed normals, but how to pre-compute normals is not elaborated in their paper.

Wildcat [6] fits ellipsoids based on the coordinates and timestamps of the clustered points. The ellipsoids representing the distribution of points are further used to generate surfels. SLICT [3] further proposes an octree-based global map and updates the distribution of points within each voxel incrementally. It obtains large-scale distributions by merging multiple voxels to generate surfels in multi-resolution.

Inspired by the above methods, we extend ikd-tree to maintain the distribution of points in each map node incrementally, and fix the distribution after its convergences.

C. LiDAR (-Inertial) Odometry

LOAM [1] has inspired many LiDAR SLAM systems due to the low coupling of system modules and the rational use of point cloud geometry attributes. However, the lack of effective map management and the high time consumption required for optimization can degrade the performance of the system.

LIO-SAM [13] proposes a framework based on keyframes and local maps, which optimizes poses in a factor graph. However, LIO-SAM builds sub-maps for input scans by simply merging point cloud of surrounding keyframes, which is a time-consuming process when the number of points is large compared to incrementally maintaining maps.

FAST-LIO [9] employs point-to-plane correspondence and the iEKF to directly register the LiDAR scan and the map. It presents a new formula to compute the Kalman gain, and the computation load only depends on the dimension of state dimension. FAST-LIO2 [10] maintains the map by an incremental kdtree data structure, namely ikd-tree, to further improve efficiency.

In this paper, we adopt the iEKF to optimize poses by directly registering the LiDAR scan and the map, but with a different data association scheme. By incorporating real-time normal and distribution of points estimation, we can efficiently construct surfels in multi-scale, which represent the local surface geometry more accurately than other geometric primitives, e.g., plane. We prioritize associating large-scale surfels over small-scale surfels since large-scale surfels are modeled with more points and are insensitive to noise.

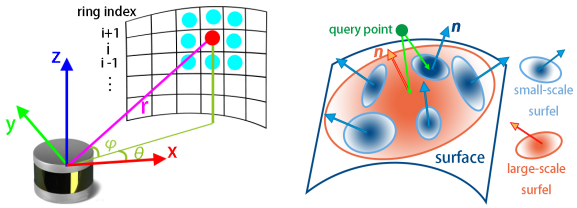
III. PRELIMINARY

A. Notation

We now define notations and frames that we used throughout the paper. We consider $\mathcal{W}$ as the world frame and $\mathcal{I}_k$, $\mathcal{L}_k$ as the IMU and LiDAR frames, related to the k-th LiDAR scan at time t_k, respectively. ${}^{a}\mathbf{T}_b \in SE(3)$ to be Euclidean transformation take 3D points from frame b to frame a, which is consisted of rotation ${}^{a}\mathbf{R}_b \in SO(3)$ and translation ${}^{a}\mathbf{t}_b \in \mathbb{R}^3$. $\boldsymbol{n}_r$ denotes the normal from Ring FALS and $\boldsymbol{e}_d$ is the eigenvector corresponding to the smallest eigenvalue of a distribution (see Section III-E).

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

B. LiDAR Observation Model



(a) LiDAR observation model (b) multi-scale surfel association

Fig. 1. Illustration of the LiDAR observation model and multi-scale surfel association. (a) The magenta line indicates the ray of the red point. The eight points are the neighborhoods that Ring FALS uses to estimate the normal of the red point. (b) The orange ellipses represent the large-scale surfel merged by the five blue small-scale surfels.

In practice, LiDAR obtains the 3D coordinates of a point by combining bearing and range measurements of the target surface [14], [15], as shown in Figure 1(a). The LiDAR observation model is as follows:

$$\boldsymbol{p}_i = r_i \boldsymbol{v}_i = r_i \begin{bmatrix} \cos\theta_i \cos\varphi_i \\ \sin\theta_i \cos\varphi_i \\ \sin\varphi_i \end{bmatrix} \tag{1}$$

where r_i is the range, θ_i the azimuth and φ_i the vertical angle of the target point. $\boldsymbol{v}_i$ represents the horizontal and vertical structural information of the point relative to the LiDAR.

For a spinning LiDAR, we denote the horizontal resolution as $H_{res} = 2\pi/m$, where m is the constant number of points within each ring. Denoting the structural information $\boldsymbol{s}_i = [\cos\theta_i \ \ \sin\theta_i \ \ \cos\varphi_i \ \ \sin\varphi_i]^T$ and then $\boldsymbol{s}_i$ can be arranged into a lookup table $\mathcal{T}$ based on the ring index and azimuth relative to the LiDAR as follows:

$$\boldsymbol{s}_i = \mathcal{T}(row_i, \ col_i) \tag{2}$$

where row_i represents the ring index of $\boldsymbol{p}_i$ and $col_i = round(\theta_i / H_{res})$.

C. Least Squares Normal Estimation

Given a subset of n 3D points $\boldsymbol{p}_i$, $i = 1, 2, ..., n$ of the surface, least squares finds the normal vector $\boldsymbol{n} = (n_x, n_y, n_z)$ and the scalar d that minimizes Equation (3).

$$e = \sum_{i=1}^{n} (\boldsymbol{p}_i^T \boldsymbol{n} - d)^2 \tag{3}$$

The closed form solution of the normal $\boldsymbol{n}$ is the eigenvector corresponding to the smallest eigenvalue of the covariance matrix in Equation (4).

$$\boldsymbol{M} = \sum_{i=1}^{n} (\boldsymbol{p}_i - \overline{\boldsymbol{p}})(\boldsymbol{p}_i - \overline{\boldsymbol{p}})^T \tag{4}$$

with $\overline{\boldsymbol{p}} = 1/n \sum_{i=1}^{n} \boldsymbol{p}_i$.

D. Distribution of Points

The distribution of points within a voxel can be represented by its mean position $\overline{\boldsymbol{p}}$ and the covariance matrix $\boldsymbol{M}$, as shown in Equation (4). And $\boldsymbol{M}$ can be further simplified as follows:

$$\boldsymbol{M} = \mathcal{S}_n - \frac{1}{n} \mathcal{P}_n \mathcal{P}_n^T \tag{5}$$

where $\mathcal{S}_n$ denotes $\sum_{i=1}^{n} \boldsymbol{p}_i \boldsymbol{p}_i^T$ and $\mathcal{P}_n$ denotes $\sum_{i=1}^{n} \boldsymbol{p}_i$. Due to the symmetric nature of $\boldsymbol{M}$, it is only necessary to record the six elements in its upper right corner.

The accurate representation of the distribution of points requires a large number of points. Due to limited resolution and occlusion, point cloud from multiple locations must be accumulated incrementally to obtain high quality maps. For newly incorporated m points in a voxel, their $\mathcal{S}_m$, $\mathcal{P}_m$ need to be calculated. Subsequently, the distribution of points within this voxel can be updated by [16]:

$$\begin{aligned} \overline{\boldsymbol{p}} &= (\mathcal{P}_n + \mathcal{P}_m)/(n+m) \\ \boldsymbol{M} &= \mathcal{S}_n + \mathcal{S}_m - \tfrac{1}{n+m}(\mathcal{P}_n + \mathcal{P}_m)(\mathcal{P}_n + \mathcal{P}_m)^T \end{aligned} \tag{6}$$

E. Surfel

We define the planarity ρ within a voxel similar to SLICT [3], and further introduce γ as following:

$$\begin{aligned} \rho_i &= 2(\lambda_2 - \lambda_1)/(\lambda_1 + \lambda_2 + \lambda_3) \\ \gamma_i &= \lambda_2 / \lambda_1 \end{aligned} \tag{7}$$

where $\lambda_1, \lambda_2, \lambda_3$ are the eigenvalues of covariance matrix $\boldsymbol{M}$ with $\lambda_1 < \lambda_2 < \lambda_3$. We define a surfel has ρ_i greater than 1.0 and γ_i greater than 100. A larger ρ_i implies that the distribution of the sampled points is flatter on the surface, and a larger γ_i indicates that the distribution is less close to a linear geometry. If the above criteria are satisfied, the surfel is represented by the mean position of points $\overline{\boldsymbol{p}}$ and the normal $\boldsymbol{e}_d$, where $\boldsymbol{e}_d$ is the eigenvector corresponding to λ_1. Multiple small-scale surfels can be merged into a large-scale surfel by merging the distributions following Equation (6). And the merged distribution still needs to satisfy the criteria in the above to be considered as a large-scale surfel.

IV. Ring FALS Normal Estimator

We first revisit FALS [7]. In FALS, Equation (3) is reformulated to obtain:

$$\widetilde{e} = \sum_{i=1}^{n} (\boldsymbol{p}_i^T \widetilde{\boldsymbol{n}} - 1)^2 \tag{8}$$

where $\widetilde{\boldsymbol{n}}$ is defined up to a scale factor. Substituting Equation (1) gives:

$$\widetilde{e} = \sum_{i=1}^{n} r_i^2 (\boldsymbol{v}_i^T \widetilde{\boldsymbol{n}} - r_i^{-1})^2 \tag{9}$$

where r_i is the range and $\boldsymbol{v}_i$ implies the bearing information of the target points related to the LiDAR.

It can be assumed that the range of points within a small region are similar, thanks to the high-resolution LiDAR.

Therefore, r_i^2 can be removed from Equation (9) to obtain an approximation:

$$\widehat{e} = \sum_{i=1}^{n} (\boldsymbol{v}_i^T \widehat{\boldsymbol{n}} - r_i^{-1})^2 \tag{10}$$

where $\widehat{\boldsymbol{n}}$ is the approximate normal, and it has the closed form solution $\widehat{\boldsymbol{n}} = \widehat{\boldsymbol{M}}^{-1}\widehat{\boldsymbol{b}}$ where $\widehat{\boldsymbol{M}} = \sum_{i=1}^{n} \boldsymbol{v}_i \boldsymbol{v}_i^T$ and $\widehat{\boldsymbol{b}} = \sum_{i=1}^{n} \boldsymbol{v}_i / r_i$. The matrix $\widehat{\boldsymbol{M}}^{-1}$ depends only on the constant structural information $\boldsymbol{v}$, independent of the range $\boldsymbol{r}$. Hence, the matrix $\widehat{\boldsymbol{M}}^{-1}$ can be pre-computed as a lookup table.

To obtain the neighborhood of $\boldsymbol{v}_i$ for computing $\widehat{\boldsymbol{M}}^{-1}$ and $\widehat{\boldsymbol{b}}$, FALS projects the LiDAR scan points onto an SRI following Equation (1). The computation of $\widehat{\boldsymbol{M}}^{-1}$ and $\widehat{\boldsymbol{b}}$ requires that each pixel in the SRI has a corresponding measurement, hence an interpolation is needed since the LiDAR scan points only occupy sparse pixels. This is in turn a time-consuming process.

Different from FALS, Ring FALS establishes a fast mapping following Equation (2) based on the structural information of the LiDAR. To avoid the costly vacant pixel interpolation, we create a table with the number of rows corresponding to the number of rings and the number of columns matching the number of points within each ring. The table is created solely based on the provided measurements, eliminating the necessity for interpolation. Thus Ring FALS speeds up the projection step in FALS and circumvents the time-consuming neighborhood search in many LIO systems, facilitating dense normal estimation for LiDAR scans.

Note that there are instances where the assumption of Ring FALS may not hold. Such cases include scenarios like wall edges, occlusions where the range of points varies significantly in a small area, and situations with missing range measurements. To address this, we flip all the backfaced normals and then apply image median blurring to smooth the normals and enhance their consistency. For points whose normal direction still differs significantly from the associated map points, we identify them as outliers in the optimization process through visibility and consistency checks, as elaborated in Section V-B2.

V. System Description

The pipeline of LOG-LIO is shown in Figure 2. For a new input scan, we first estimate the normal of the points. The association is then performed between the undistorted point cloud and the map according to their local geometric information. We incorporate the measurements of IMU and optimize the poses of the body via iEKF. After optimization, new points are added to the map managed by the extended ikd-tree, and the distribution within a voxel is incrementally maintained.

Our system takes the IMU frame as the body frame, where the system state $\mathbf{x}$ can be written as:

$$\mathbf{x} = \begin{bmatrix} {}^{\mathcal{W}}\mathbf{R}_{\mathcal{I}}^T & {}^{\mathcal{W}}\mathbf{p}_{\mathcal{I}}^T & {}^{\mathcal{W}}\mathbf{v}_{\mathcal{I}}^T & \mathbf{b}_{\omega}^T & \mathbf{b}_{a}^T & {}^{\mathcal{W}}\mathbf{g}^T \end{bmatrix} \tag{11}$$

where ${}^{\mathcal{W}}\mathbf{R}_{\mathcal{I}}^T$, ${}^{\mathcal{W}}\mathbf{p}_{\mathcal{I}}^T$ and ${}^{\mathcal{W}}\mathbf{v}_{\mathcal{I}}^T$ are the orientation, position and velocity of IMU in the world frame (i.e., the first IMU frame), $\mathbf{b}_{\omega}^T$ and $\mathbf{b}_{a}^T$ are gyroscope and accelerometer bias respectively, ${}^{\mathcal{W}}\mathbf{g}^T$ is the known gravity vector in the world frame.

A. Data Pre-processing

LOG-LIO uses Ring FALS (Section IV) to to estimate the normal for each input point, denoted as $\boldsymbol{n}_r$. Subsequently, voxel grid downsampling and backward propagation based on IMU measurements are used for point reduction and distortion correction, respectively.

B. Data Association

At the beginning of data association, the IMU measurements are integrated from the previous frame to predict the pose $\widehat{\mathbf{x}}_k$. Using this prediction, each new input point ${}^{\mathcal{L}}\boldsymbol{p}_i$ is transformed to the world frame ${}^{\mathcal{W}}\boldsymbol{p}_i = {}^{\mathcal{W}}\widehat{\boldsymbol{T}}_{\mathcal{I}} \, {}^{\mathcal{I}}\boldsymbol{T}_{\mathcal{L}} \, {}^{\mathcal{L}}\boldsymbol{p}_i$. Then, data association is performed in three consecutive steps:

1) Initial Correspondence: For a query point ${}^{\mathcal{W}}\boldsymbol{p}_i$, we first search for its k nearest map points corresponding to k nearest map voxels.

2) Visibility and Consistency Checks: The candidate associated map points may not be visible to the LiDAR if the angle between the normal of the map point and the ray (vector from the query scan point to the LiDAR center) is greater than 90 degrees. Such a case usually occurs indoors, where the two planes of an object (e.g., a wall) are close to each other, which is referred to as the double-side issue [17]. This incorrect correspondence is eliminated directly.

The consistency of the associated map points is evaluated by computing the average angle between the normal of the query point and the normals of the associated map points. If the average angle is larger than a threshold α ($\alpha = 60°$), we consider it an inconsistent association and discard it.

3) Hierarchical Association: For query points satisfying visibility and consistency checks, a hierarchical association is performed, where point-to-surfel is prioritized over point-to-plane, and large-scale surfels are prioritized over small-scale surfels.

Surfels offer a more precise and flexible representation of a local surface compared to a plane fitted with sparse map points since they are modeled with the distribution of points, which not only indicates the location but also captures the shape of the local surface. Large-scale surfel can be approximated by merging multiple small-scale distributions following Equation (6). Moreover, they exhibit a high tolerance to noise, thereby providing more robust constraints when contrasted with small-scale surfels. As illustrated in Figure 1(b), the orange ellipse depicts the large-scale surfel merged by the five blue small-scale surfels. The green query point is initially associated with the merged large-scale surfel if the merged large-scale surfel satisfies the criteria outlined in Section III-E, and the distance from the mean position of each small-scale surfel to the large-scale surfel is below a predefined threshold. Otherwise, the association with small-scale surfels is preferred.

For constraints with small-scale surfels, we associate the query point with the surfel of the voxel where the point is located, which must already be fixed. If the voxel cannot meet the criteria of a surfel (Section III-E), we resort to using the point-to-plane association as LOAM [1].

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

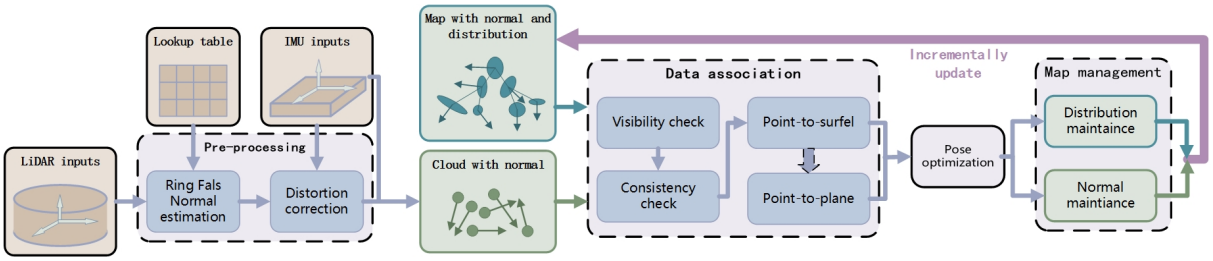


Fig. 2. System overview of LOG-LIO

C. Pose optimization

We adopt the iEKF from FAST-LIO2 to optimize the pose. The prediction step is implemented by the integration of IMU measurements from the latest optimized state $\overline{\mathbf{x}}_{k-1}$ along with the covariance matrix $\overline{\mathbf{P}}_{k-1}$.

For the residual computation, given a point ${}^{\mathcal{W}}\boldsymbol{p}_i$ in the world frame, the residual z_i is calculated as:

$$\mathbf{z}_i = \boldsymbol{n}_j({}^{\mathcal{W}}\boldsymbol{p}_i - {}^{\mathcal{W}}\boldsymbol{q}_j) \tag{12}$$

where $\boldsymbol{n}_j$ is the normalized normal of the associated surfel or plane for $\boldsymbol{p}_i$, and ${}^{\mathcal{W}}\boldsymbol{q}_j$ is a point lying on the associated element.

Then, we denote the propagated state and covariance by $\widehat{\mathbf{x}}_k$ and $\widehat{\mathbf{P}}_k$ respectively. They represent the prior Gaussian distribution for the state. By incorporating the prior distribution and the measurement models for point-to-surfel and point-to-plane associations from Equation (12), we obtain the maximum a-posterior estimate (MAP) as follows:

$$\begin{aligned} \min_{\widetilde{\mathbf{x}}_k^\kappa}(&\|\mathbf{x}_k \boxminus \widehat{\mathbf{x}}_k\|^2_{\widehat{\mathbf{P}}_k} + \sum_{i\in surfel} \|\mathbf{z}_i^\kappa + \mathbf{H}_i^\kappa \widetilde{\mathbf{x}}_k^\kappa\|^2_{\mathbf{R}_i} \\ &+ \sum_{j\in plane} \|\mathbf{z}_j^\kappa + \mathbf{H}_j^\kappa \widetilde{\mathbf{x}}_k^\kappa\|^2_{\mathbf{Q}_j}) \end{aligned} \tag{13}$$

where $\boxminus$ computes the difference between $\mathbf{x}_k$ and $\widehat{\mathbf{x}}_k$ in the local tangent space of $\mathbf{x}_k$, $\widetilde{\mathbf{x}}_k^\kappa$ is the error of the κ-th iterate update at time k, $\mathbf{H}_i^\kappa$ and $\mathbf{H}_j^\kappa$ are Jacobian matrices with respect to $\widetilde{\mathbf{x}}_k^\kappa$, $\mathbf{R}_i$ and $\mathbf{Q}_j$ come from the raw measurement noise. Compared with FAST-LIO2, we augment the MAP with point-to-surfel associations, which are the middle term of Equation (13). The Kalman gain can be computed efficiently, with the computation load depending on the state dimension instead of the measurement dimension [9], [10].

D. Map Management

LOG-LIO uses an extended ikd-tree to manage the map. The ikd-tree originally stores map points in both leaf nodes and internal nodes [10]. In our extension, we additionally store a distribution in each node, and this distribution is maintained through voxels. Upon the first scan, we initialize the tree-structured map with a predetermined voxel resolution and associate the distribution of points within each voxel with the corresponding tree node. For subsequent points, if they fall within the same voxel as the nearest associated map point, we incrementally update the distribution within the voxel (Section III-D). In cases where the points belong to a different voxel, we initialize a new tree node encompassing those points and the voxel and add it to the map.

To balance computational efficiency and accuracy, we limit the number of points added to each voxel by tuning the down-sampling rate in pre-processing. Additionally, we consider the distribution stabilizes once the directions of $\boldsymbol{n}_r$ and $\boldsymbol{e}_d$ converge. The distribution is then fixed in the map.

VI. EXPERIMENTAL RESULTS

A. Implementation details

In data pre-processing, we set the downsampling grid to match the map's voxel size, ensuring that each voxel contains at most one point per frame. And we normalize the normal after downsamping. Within the extended ikd-tree nodes, we maintain the point distribution, updating it once the voxel accumulates $\eta = 25$ points. When the angle between $\boldsymbol{n}_r$ and $\boldsymbol{e}_d$ falls below 20 degrees, we consider the distribution stabilized, and both $\boldsymbol{n}_r$ and $\boldsymbol{e}_d$ are fixed and replaced by the average of their values. For map voxels that accumulate 2η points without stabilization, we conclude that their distribution no longer requires updates and fix it.

B. Experimental Settings

The experiment focuses on the following two research questions:

- Can Ring FALS estimates the normal of LiDAR points in real-time and accurately represents environmental information?
- Can LOG-LIO improve the accuracy of pose estimation by incorporating normal and distribution of points estimation?

We conduct extensive experiments on the M2DGR [18] and NTU VIRAL [19] datasets, both of which include 9-axis IMU measurements and ground truth trajectories. The M2DGR dataset collects data on a ground platform equipped with Velodyne-32 LiDAR and captured in diverse indoor and outdoor scenarios with ground truth trajectories obtained from laser 3D tracking, motion capture, and RTK receivers. The NTU VIRAL dataset collects data on an Unmanned Aerial Vehicle (UAV) platform with ground truth obtained by a laser-tracker total station with centimeter-level accuracy. The horizontal Ouster 16-channel OS1 LiDAR and VectorNav VN100 IMU are used. Compared with M2DGR, the LiDAR used by VIRAL has a sparser point cloud, making it more challenging to estimate poses in open areas. In the experiments, the

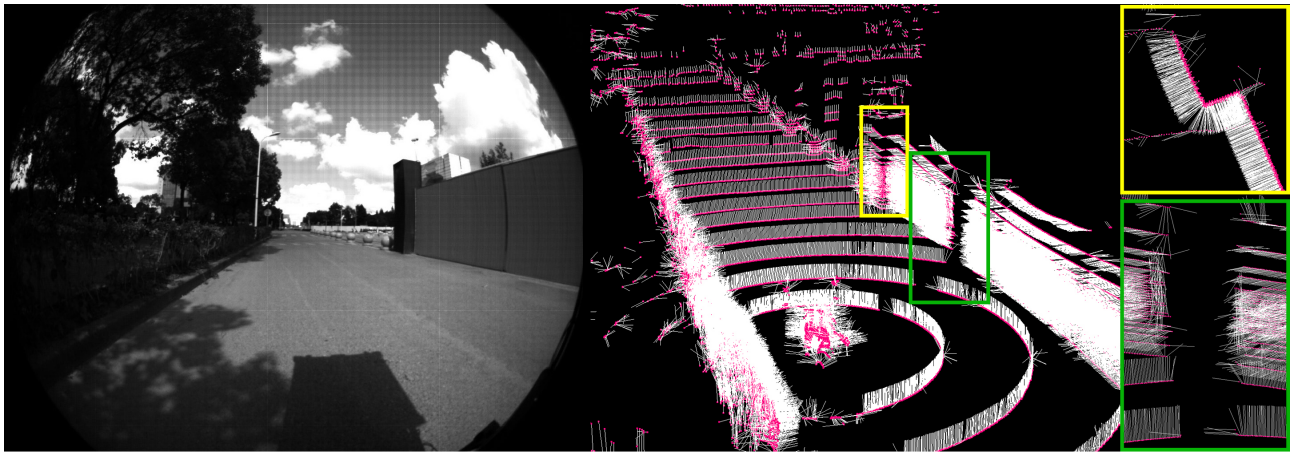


Fig. 3. The starting position of the sequence *gate03* of M2DGR dataset. The white lines represent normalized normals from Ring FALS estimation.

TABLE I
THE MEAN RUNNING TIME(MS) OF NORMAL ESTIMATION FOR A SINGLE SCAN TO CERTAIN LIDARS

	points	Ring FALS				PCL	
		projection	box-filtering	smoothing	total	single thread	OMP 10 threads
Velodyne-32	57600	2.045	2.540	3.199	**7.784**	79.811	26.355
Ouster-16	16384	0.560	1.221	0.815	**2.597**	155.972	39.664

resolution of maps and new scan downsampling size are set to 0.4 m for M2DGR and 0.5 m for NTU VIRAL respectively. Our workstation runs with Ubuntu 18.04, equipped with an Intel Core Xeon(R) Gold 6248R 3.00GHz processor and 32GB RAM.

C. Evaluation of Normal Estimation

We conduct a comparative analysis of Ring FALS and PCL [11] normal estimation tools. The implementation of PCL normal estimation is based on traditional least squares with the assistance of kdtree. PCL also provides a parallel implementation using OpenMP to speed up the computation. Note that normal smoothing is a time-consuming process for PCL, so only Ring FALS smoothness the normals.

To provide an intuitive evaluation, we visualize the normals at the starting position of sequence *gate03* of the M2DGR dataset. As shown in Figure 3, the white lines represent the normalized normals estimated by Ring FALS. Notably, almost all ground point normals exhibit a vertical upward orientation. With respect to the pillar in the yellow box, the normals at the corners transit smoothly with the normals on the adjacent sides. Due to occlusion, points near fake edges within the green box fail to meet the neighborhood range similarity assumption, leading to inaccurate Ring FALS estimation. However, these points with misestimated normals are a minority within the scan and are filtered out during visibility and consistency checks (Section V-B2).

Table I shows the average processing times of normal estimation for a single LiDAR scan from the M2DGR and NTU VIRAL datasets, respectively. The M2DGR dataset contains about 57,600 points per scan. Ring FALS demonstrates significantly reduced processing time compared to PCL, taking only one-tenth of the time, and it's four times faster than the OpenMP version. For NTU VIRAL, Ring FALS also achieves significantly shorter processing times compared to PCL, with or without OpenMP.

It is noteworthy that despite the Ouster-16 LiDAR having fewer points than Velodyne-32 in one scan, the consumption time increases. This is due to the time-consuming kdtree neighborhood search in PCL's normal estimation. And it is influenced by the spatial structure of the kdtree, which, in turn, reflects the complexity of the environment.

D. Evaluation of Odometry

We compare LOG-LIO with two state-of-the-art LIO methods, FAST-LIO2 [10] and LIO-SAM [20] without enabling loop closure. Accuracy assessment is based on the root-mean-square error (RMSE) of absolute trajectory error (ATE). We employ LOG-C, which only performs point-to-plane association, for ablation experiments.

1) M2DGR Datasets: Due to the instability of the RTK signal, the first 100 seconds and the last 100 seconds of *street07* and *street10* are discarded in the experiment.

Table II reports the quantitative results. Notably, LOG-LIO, LOG-C, and FAST-LIO2 show comparable accuracy in indoor scenes, such as *doors* and *halls*, outperforming LIO-SAM in most cases. This is due to the abundance of planar features in indoor scenes, which results in more map points forming true planes. Consequently, the point-to-plane data association provides effective constraints for pose estimation. Conversely, the point-to-line data association of LIO-SAM may become less reliable, especially when errors accumulate.

In outdoor sequences, i.e., *gate*, *street*, map points are relatively sparse compared to indoor scenes. This is especially

evident in the *street* sequences, where the robot moves on wide campus roads at night. Figure 4 shows the trajectories of sequence *street10* for qualitative comparison. The competitive results of LOG-LIO suggest that efficient and accurate estimation of local geometric information exhibits great potential in reducing the error of LIO system.

Overall, when employing only point-to-plane association, LOG-C demonstrates a slight improvement in average accuracy when compared to FAST-LIO2. However, with the implementation of our proposed hierarchical data association and map management scheme, LOG-LIO consistently achieves the lowest mean error and gets the best results in 10 out of the 21 sequences.

TABLE II
THE TRANSLATION RMSE(M) RESULTS OF POSE ESTIMATION COMPARISON ON THE M2DGR DATASET

Seq	duration(s)	LOG-LIO	LOG-C	FAST-LIO2	LIO-SAM
gate01	172	0.097	**0.085**	0.091	0.122
gate02	327	**0.270**	0.277	0.279	0.288
gate03	283	**0.085**	**0.085**	0.109	0.095
walk01	291	0.078	**0.077**	0.112	0.080
door01	461	0.251	**0.249**	0.271	0.269
door02	127	**0.172**	0.177	0.200	0.180
street01	1028	**0.246**	0.294	0.329	0.559
street02	1227	2.448	**2.228**	2.754	3.320
street03	354	**0.097**	0.103	0.106	0.102
street04	858	**0.485**	0.565	0.552	1.009
street05	469	0.331	**0.287**	0.377	0.407
street06	494	0.342	0.391	0.434	**0.332**
street07	829	2.916	3.646	3.512	**1.614**
street08	491	**0.130**	0.146	0.170	0.161
street09	907	3.164	3.754	3.648	**2.657**
street10	810	**0.388**	0.977	0.956	8.560
hall01	351	**0.256**	**0.256**	0.258	0.281
hall02	128	0.274	**0.272**	0.274	0.285
hall03	164	0.345	0.359	**0.343**	0.579
hall04	181	**0.944**	**0.944**	0.952	1.076
hall05	402	1.045	1.046	1.049	**1.015**
mean		**0.684**	0.772	0.799	1.095

The best and second-best results are bolded and underlined respectively.

2) NTU VIRAL Datasets: As depicted in Table III, LOG-LIO demonstrates the best performance in most sequences, closely followed by LOG-C. LIO-SAM achieves the best results on several sequences but fails in half of the dataset.

In the sequences *nya* and *tnp*, where the drone traverses indoors, LOG-LIO, LOG-C and FAST-LIO2 exhibit similar errors. This behavior is consistent with what we observed in M2DGR, where the presence of numerous planes in confined spaces can effectively constrain the point-to-plane association. In the *eee*, *sbs*, and *rtp* outdoor scenes amidst buildings, LOG-LIO, LOG-C, and FAST-LIO2 yield highly similar trajectories due to effective constraints imposed by the plane structure. However, in the *spms* sequences, where the drone departs from an area surrounded by buildings and ascends to higher altitudes, the sparse LiDAR points lead to limited map overlap. This can potentially result in registration errors when performing point-to-plane data association. LOG-LIO addresses this challenge by performing point association with corresponding voxels. The accurate local geometric information within these voxels help mitigate registration errors, ultimately resulting in more precise trajectories.

It is worth noting that in practice, trajectory error in the LIO system should consider various factors such as map resolution, IMU noise, and etc. And we focus on factors closely tied to our contributions while keeping the other parameters fixed in this paper.

TABLE III
THE TRANSLATION RMSE(M) RESULTS OF POSE ESTIMATION COMPARISON ON THE NTU VIRAL DATASET

Seq	duration(s)	LOG-LIO	LOG-C	FAST-LIO2	LIO-SAM
eee_01	399	0.084	0.082	0.084	**0.049**
eee_02	321	0.072	0.072	0.073	**0.051**
eee_03	181	0.112	0.114	0.113	**0.081**
nya_01	396	0.081	0.080	**0.075**	0.174
nya_02	428	0.111	0.111	0.109	**0.085**
nya_03	411	**0.120**	0.121	0.121	0.249
sbs_01	354	**0.094**	0.095	0.097	x
sbs_02	373	0.085	0.086	**0.080**	**0.080**
sbs_03	389	0.085	**0.083**	**0.083**	x
rtp_01	482	**0.191**	0.230	0.209	x
rtp_02	453	0.147	0.153	0.163	**0.117**
rtp_03	355	0.180	0.181	0.170	**0.113**
tnp_01	583	**0.093**	0.095	0.094	x
tnp_02	457	0.075	**0.058**	0.071	x
tnp_03	407	0.084	**0.079**	0.080	x
spms_01	446	**1.450**	1.670	1.818	x
spms_02	398	**2.196**	2.748	3.378	x
spms_03	386	**0.685**	0.766	0.793	x
mean		**0.330**	0.379	0.423	x

The best and second-best results are bolded and underlined respectively.

3) Processing Time Evaluation: We perform statistical analysis on the time consumption of LOG-LIO and FAST-LIO2 in each sequence, as shown in Table IV. It is observed that the average processing time per scan of LOG-LIO is slightly longer than that of FAST-LIO2, which is mainly due to the Ring FALS normal estimation and incremental point distribution maintenance within map voxels. Despite LOG-LIO exhibiting an additional average time consumption of 8 ms than FAST-LIO2, it still meets real-time requirements. This performance difference should be considered in light of the number of points and the complexity of the environment.

VII. CONCLUSION AND FUTURE WORK

This paper introduces LOG-LIO, an online LiDAR-inertial odometry method that estimates normal and distribution of points for local geometric information in real time. To improve normal estimation efficiency for LiDAR scans, we introduce Ring FALS, an efficient normal estimator that pre-records structural information and uses range data for normal estimation. In LOG-LIO, we manage the map using an extended ikd-tree, incrementally maintaining normal and point distribution within map voxels. We employ a hierarchical data association scheme for accurate constraints, resulting in precise pose estimation. Experimental results show that LOG-LIO is competitive with state-of-the-art LIO systems in various environments.

For future research, we intend to incorporate dynamic noise removal and loop closure to enhance stability in dynamic environments and ensure long-term operation.

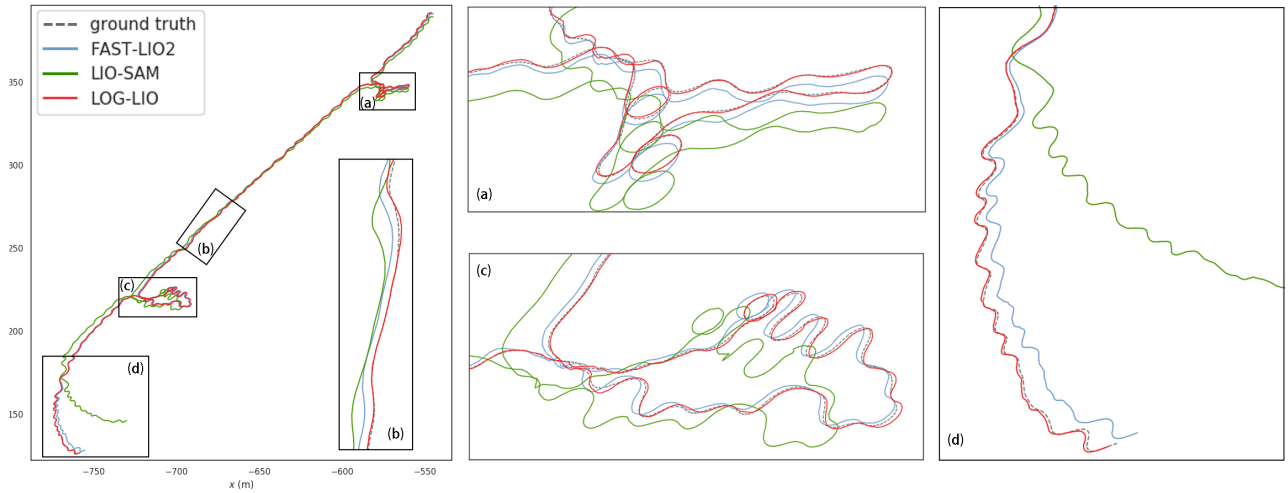


Fig. 4. Localization estimates in sequence *street10* of the M2DGR dataset. The zoomed in image of the colored boxes corresponds to the boxes of the same color in the trajectory.

TABLE IV
THE AVERAGE TIME CONSUMPTION(MS) OF EACH SEQUENCE IN THE EXPERIMENTS

	M2DGR					NTU VIRAL						
	gate	walk	door	street	hall	eee	nya	sbs	rtp	tnp	spms	mean
LOG-LIO	46.563	45.964	24.083	42.480	24.903	20.991	18.027	17.997	25.388	19.412	23.348	28.101
FAST-LIO2	31.378	32.276	14.754	28.970	15.509	15.705	12.476	12.785	21.006	13.340	17.106	**20.523**

REFERENCES

[1] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time." in *Robotics: Science and Systems*, vol. 2, no. 9. Berkeley, CA, 2014, pp. 1–9.

[2] K. Chen, B. T. Lopez, A.-a. Agha-mohammadi, and A. Mehta, "Direct lidar odometry: Fast localization with dense point clouds," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2000–2007, 2022.

[3] T.-M. Nguyen, D. Duberg, P. Jensfelt, S. Yuan, and L. Xie, "Slict: Multi-input multi-scale surfel-based lidar-inertial continuous-time odometry and mapping," *IEEE Robotics and Automation Letters*, vol. 8, no. 4, pp. 2102–2109, 2023.

[4] A. Reinke, M. Palieri, B. Morrell, Y. Chang, K. Ebadi, L. Carlone, and A.-A. Agha-Mohammadi, "Locus 2.0: Robust and computationally efficient lidar odometry for real-time 3d mapping," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 9043–9050, 2022.

[5] M. Palieri, B. Morrell, A. Thakur, K. Ebadi, J. Nash, A. Chatterjee, C. Kanellakis, L. Carlone, C. Guaragnella, and A.-a. Agha-Mohammadi, "Locus: A multi-sensor lidar-centric solution for high-precision odometry and 3d mapping in real-time," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 421–428, 2020.

[6] M. Ramezani, K. Khosoussi, G. Catt, P. Moghadam, J. Williams, P. Borges, F. Pauling, and N. Kottege, "Wildcat: Online continuous-time 3d lidar-inertial slam," *arXiv preprint arXiv:2205.12595*, 2022.

[7] H. Badino, D. Huber, Y. Park, and T. Kanade, "Fast and accurate computation of surface normals from range images," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 3084–3091.

[8] R. Fan, H. Wang, B. Xue, H. Huang, Y. Wang, M. Liu, and I. Pitas, "Three-filters-to-normal: An accurate and ultrafast surface normal estimator," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5405–5412, 2021.

[9] W. Xu and F. Zhang, "Fast-lio: A fast, robust lidar-inertial odometry package by tightly-coupled iterated kalman filter," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3317–3324, 2021.

[10] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "Fast-lio2: Fast direct lidar-inertial odometry," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2053–2073, 2022.

[11] R. B. Rusu and S. Cousins, "3d is here: Point cloud library (pcl)," in *2011 IEEE international conference on robotics and automation*. IEEE, 2011, pp. 1–4.

[12] A. Segal, D. Haehnel, and S. Thrun, "Generalized-icp." in *Robotics: science and systems*, vol. 2, no. 4. Seattle, WA, 2009, p. 435.

[13] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "isam2: Incremental smoothing and mapping using the bayes tree," *The International Journal of Robotics Research*, vol. 31, no. 2, pp. 216–235, 2012.

[14] C. Yuan, X. Liu, X. Hong, and F. Zhang, "Pixel-level extrinsic self calibration of high resolution lidar and camera in targetless environments," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7517–7524, 2021.

[15] C. Yuan, W. Xu, X. Liu, X. Hong, and F. Zhang, "Efficient and probabilistic adaptive voxel mapping for accurate online lidar odometry," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 8518–8525, 2022.

[16] S. Thrun, "Probabilistic robotics," *Communications of the ACM*, vol. 45, no. 3, pp. 52–57, 2002.

[17] L. Zhou, D. Koppel, and M. Kaess, "Lidar slam with plane adjustment for indoor environment," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7073–7080, 2021.

[18] J. Yin, A. Li, T. Li, W. Yu, and D. Zou, "M2dgr: A multi-sensor and multi-scenario slam dataset for ground robots," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2266–2273, 2021.

[19] T.-M. Nguyen, S. Yuan, M. Cao, Y. Lyu, T. H. Nguyen, and L. Xie, "Ntu viral: A visual-inertial-ranging-lidar dataset, from an aerial vehicle viewpoint," *The International Journal of Robotics Research*, vol. 41, no. 3, pp. 270–280, 2022.

[20] 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 *2020 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2020, pp. 5135–5142.