remote sensing

MDPI

Technical Note

Real-Time 3D Mapping in Complex Environments Using a Spinning Actuated LiDAR System

Li Yan [1,2], Jicheng Dai [1], Yinghao Zhao [1] and Changjun Chen [1,2,*]

[1] School of Geodesy and Geomatics, Wuhan University, Wuhan 430079, China
[2] Hubei Luojia Laboratory, Wuhan University, Wuhan 430079, China
* Correspondence: chencj@whu.edu.cn

Abstract: LiDAR is a crucial sensor for 3D environment perception. However, limited by the field of view of the LiDAR, it is sometimes difficult to achieve complete coverage of the environment with a single LiDAR. In this paper, we designed a spinning actuated LiDAR mapping system that is compatible with both UAV and backpack platforms and propose a tightly coupled laser–inertial SLAM algorithm for it. In our algorithm, edge and plane features in the point cloud are first extracted. Then, for the significant changes in the distribution of point cloud features between two adjacent scans caused by the continuous rotation of the LiDAR, we employed an adaptive scan accumulation method to improve the stability and accuracy of point cloud registration. After feature matching, the LiDAR feature factors and IMU pre-integration factor are added to the factor graph and jointly optimized to output the trajectory. In addition, an improved loop closure detection algorithm based on the Cartographer algorithm is used to reduce the drift. We conducted exhaustive experiments to evaluate the performance of the proposed algorithm in complex indoor and outdoor scenarios. The results showed that our algorithm is more accurate than the state-of-the-art algorithms LIO-SAM and FAST-LIO2 for the spinning actuated LiDAR system, and it can achieve real-time performance.

Keywords: LiDAR; point cloud; multi-sensor fusion; simultaneous localization and mapping

Citation: Yan, L.; Dai, J.; Zhao, Y.; Chen, C. Real-Time 3D Mapping in Complex Environments Using a Spinning Actuated LiDAR System. *Remote Sens.* **2023**, *15*, 963. https://doi.org/10.3390/rs15040963

Academic Editor: Jorge Delgado García

Received: 28 December 2022
Revised: 3 February 2023
Accepted: 6 February 2023
Published: 9 February 2023

1. Introduction

Light Detection and Ranging (LiDAR) can achieve highly accurate and efficient distance measurement. I is insensitive to illumination changes because it measures by actively emitting a pulsed laser. By combining LiDAR with Simultaneous Localization and Mapping (SLAM) technology, the laser SLAM system can obtain a three-dimensional map of the surrounding environment both indoors and outdoors. These advantages make the laser SLAM system play an important role in many fields, such as autonomous driving [1], building inspection [2], forestry investigation [3], etc. Currently, the LiDAR used for laser SLAM can be basically divided into two categories. Mechanical LiDAR is the most commonly used LiDAR type. Its horizontal Field Of View (FOV) can reach close to 360°, but its vertical FOV is very limited. Besides, with the development of microelectronics technology, solid-state LiDAR, e.g., the DJI Livox series, has become more and more commonly used [4,5]. Generally, solid-state LiDAR can provide a larger vertical FOV than mechanical LiDAR, but its horizontal FOV is much smaller. Therefore, in many scenarios, both types of LiDAR cannot completely cover the whole environment. This shortcoming greatly limits the mapping efficiency of laser SLAM systems, especially when using a platform with limited endurance such as an Unmanned Aerial Vehicle (UAV) to carry the mapping system [6]. In narrow environments, this limited FOV will degrade the accuracy and stability of localization as only a small number of objects can be scanned.

Much research has been conducted to expand the FOV of LiDAR, and the solutions can be roughly divided into two categories. The first category is to combine multiple LiDARs [7–11]. Such systems typically use a horizontally mounted LiDAR to provide

primary positioning information and a vertically mounted LiDAR to improve the scanning coverage. Its advantage is that the mechanical structure is simple. However, due to the high price of LiDARs, the cost of these solutions is greatly increased. Another category of solutions is to actuate the LiDAR dynamically, and LOAM [12] is one of the most-famous methods among them. LOAM first accumulates the point cloud with the assistance of the IMU and then matches the accumulated point cloud with the global map to correct the accumulated errors. Considering that the prediction error of the IMU grows exponentially with time and the feature extraction and matching algorithms need to wait for sufficient data to be accumulated, this time interval will become a bottleneck limiting the accuracy of the SLAM system.

To address this problem, we designed a mapping system based on a spinning actuated multi-beam LiDAR. By using a multi-beam LiDAR, more information can be obtained in each scan, which helps to increase the frequency of point cloud matching and generate a denser point cloud map. We first extracted feature points from each frame point cloud through an improved feature extraction method based on LOAM. Then, we judged whether the current point cloud contains enough information by analyzing the spatial distribution of feature points and performed scan-to-map matching once the requirement is met. Compared with accumulating point clouds with a fixed number of scans, this method allows a better balance between the matching frequency of point clouds and the accuracy and reliability of the matching results. Finally, in order to eliminate the accumulated error, we added a loop closure detection module to the algorithm. The main contributions of this paper can be summarized as follows:

- We propose a tightly coupled laser–inertial SLAM algorithm named Spin-LOAM for a spinning actuated LiDAR system.
- An adaptive scan accumulation method that can improve the accuracy and reliability of matching by analyzing the spatial distribution of feature points.
- Extensive experiments were conducted in indoor and outdoor environments to verify the effectiveness of our algorithm.

2. Related Work

As a fundamental problem in robotics, numerous SLAM algorithms based on LiDAR have been proposed. LeGO-LOAM [13] extracts the ground point cloud from the real-time scan results to improve the accuracy in the elevation direction. T-LOAM [14] simultaneously extracts edge features, sphere features, planar features, and ground features to improve the matching accuracy. The works [15–17] refined the localization result by introducing plane constraints. Suma++ [18] not only uses geometric features in point clouds, but also introduces semantic information to assist point cloud matching. Besides these LiDAR-only odometry algorithms, many multi-sensor-fusion-based algorithms have been proposed to improve the accuracy and robustness. LIO-SAM [19] combines the IMU pre-integration factor with the LiDAR odometry factor through a factor graph. LIO-Mapping [20] integrates the LiDAR and IMU in a tightly coupled fashion. FAST-LIO [21] adopts a tightly coupled iterated extended Kalman filter on a manifold to fuse the data and is accelerated by introducing an incremental KD-Tree in FAST-LIO2 [22]. CLINS [23] fuses LiDAR and IMU data by representing trajectories as a continuous-time function, and this framework is well compatible with arbitrary-frequency data from other asynchronous sensors.

For the purposes of acquiring complete 3D information of the environment, researchers have proposed many actuated LiDAR systems. Bosse et al. [24] designed a mapping system named Zebedee, which connects the sensors to the platform via springs. By treating the trajectory as a function of time, a surfel-based matching algorithm was adopted to estimate the 6-DOF pose. Kaul et al. [25] proposed a passively actuated rotating LiDAR system for UAV mapping, and they used a continuous-time SLAM algorithm to produce the trajectory. However, it cannot process the data in real-time. Park et al. [26] addressed this issue by introducing map deformation to replace the original global trajectory optimization in continuous-time SLAM. Fang et al. [27] proposed a two-stage matching algorithm to

estimate the trajectory of a rotating LiDAR. In the algorithm, the distortion of the current point cloud was first removed using the estimated motion generated by matching it with the local map, and then, the undistorted point cloud was matched with the global map. Unlike the aforementioned works, Zhan et al. [28] used a rotating multi-beam LiDAR for 3D mapping and combined it with stereo cameras for dense 3D reconstruction. The precision of this system is very high, but it needs to remain stationary while collecting data. R-LOAM [29] improves the localization accuracy of rotating LiDAR by leveraging prior knowledge about a reference object. Karimi et al. [30] proposed an actuated LiDAR system using the Lissajous pattern [31]. By using the scan slice instead of the full-sweep point cloud to match with the global map, they achieved low-latency localization for a UAV without an IMU in an indoor environment.

3. System Overview

We first introduce the hardware systems used in the study. As shown in Figure 1, our device mainly consists of a laser scanner, a step motor, and an IMU, in which the scanner is driven to rotate by the motor. The rotation angle is recorded by an encoder. The IMU is rigidly attached to the platform, so we regarded the IMU frame $\{I\}$ as the body frame $\{B\}$ for simplicity. The LiDAR frame is denoted as $\{L\}$, and the fixed LiDAR frame $\{FL\}$ coincides with the initial LiDAR frame when the motor is reset. The Y-axis of the motor frame $\{M\}$ is aligned with the spin axis. The two extrinsic parameters $\boldsymbol{T}_I^{FL} \in SE(3)$ and $\boldsymbol{T}_M^{FL} \in SE(3)$ are both calibrated manually, where $\boldsymbol{T}_I^{FL}$ represents the transformation from the frame $\{FL\}$ to the frame $\{I\}$ and $\boldsymbol{T}_M^{FL}$ represents the transformation from the frame $\{FL\}$ to the frame $\{M\}$. The timestamp of each sensor is synchronized at the hardware level to ensure accuracy.

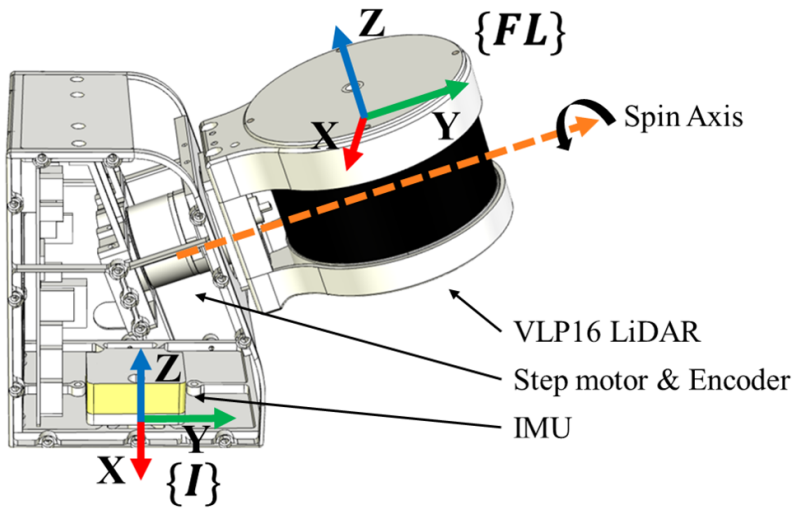


Figure 1. The mechanical structure of the hardware device.

Figure 2 provides an overview of our SLAM algorithm. In the front-end, first, the IMU measurements are used to construct the pre-integration factor and produce pose predictions. Next, the raw LiDAR point cloud is transformed to the fixed LiDAR frame and de-skewed using the pose predictions and motor encoder data. Then, edge and plane feature points are extracted from the de-skewed point cloud. In the scan-to-map registration module, the correspondences of these feature points and the global map are established. The spatial distribution of these matching point pairs is examined to decide whether they are to be added to the factor graph or accumulated to the next scan. In the back-end, the IMU pre-integration factor and LiDAR factors are jointly optimized to estimate the system states, as shown in Figure 3. In order to bound the amount of computation, only the latest state is optimized when no loop closure constraints are added. After optimization, the feature

points are added to the global map using the optimized state. Loop closure detection is performed periodically in the background to reduce drift.

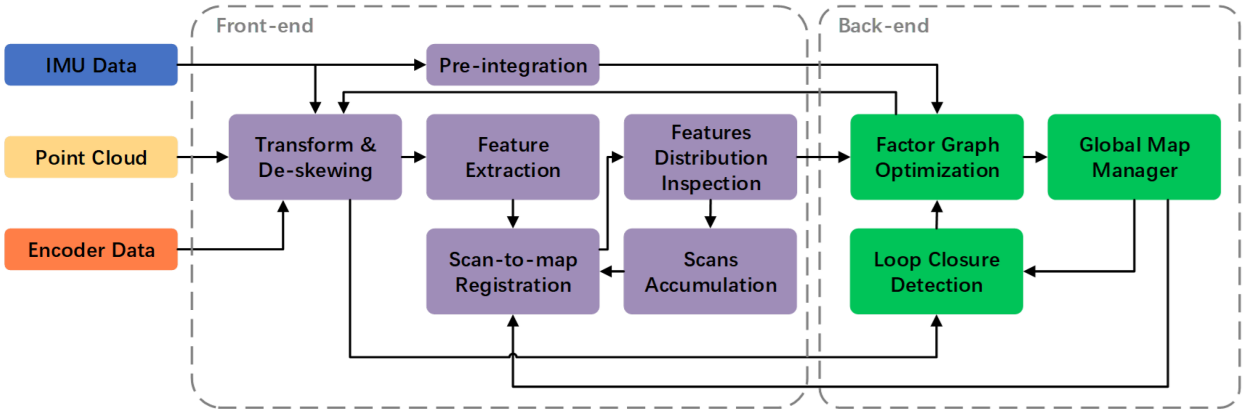


Figure 2. System overview of the proposed laser–inertial SLAM algorithm for the spinning actuated LiDAR system.

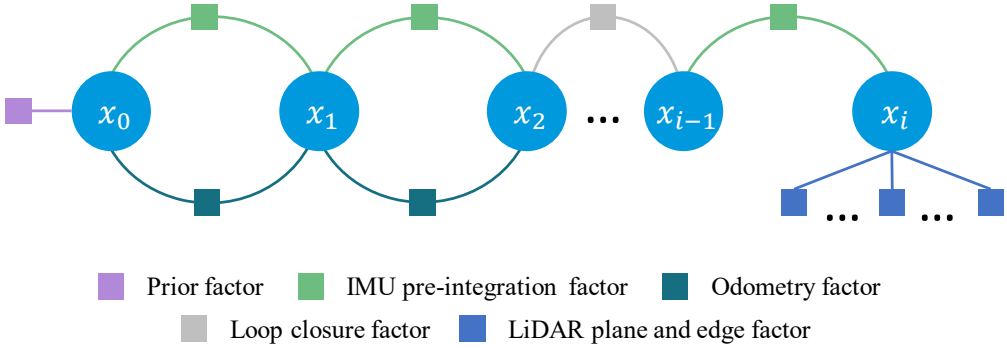


Figure 3. Structure of the factor graph. Only the LiDAR feature factor of the latest scan is directly used in the factor graph, and the others are replaced by the odometry factors, which are relative transformed between two adjacent scans.

4. Methodology

The system state $\boldsymbol{x}$ to be estimated at time t is defined as

$$\boldsymbol{x}_t = (\boldsymbol{R}_t, \boldsymbol{p}_t, \boldsymbol{v}_t, \boldsymbol{b}_t^{\omega}, \boldsymbol{b}_t^{a}) \tag{1}$$

where $\boldsymbol{b}_t^{\omega} \in \mathbb{R}^3$ and $\boldsymbol{b}_t^{a} \in \mathbb{R}^3$ are the gyroscope and accelerometer biases, respectively. $\boldsymbol{R}_t \in SO(3)$, $\boldsymbol{p}_t \in \mathbb{R}^3$, and $\boldsymbol{v}_t \in \mathbb{R}^3$ are the orientation, position, and velocity of the sensor platform in the global coordinate frame $\{G\}$, respectively.

4.1. IMU Processing

The platform's angular velocity $\boldsymbol{\omega}_t$ and acceleration $\boldsymbol{a}_t$ in the IMU frame can be obtained by the IMU, but the raw measurements of the IMU are corrupted by noise and bias. Commonly, the slow variations in the bias are modeled with Brownian motion; hence, the IMU measurement model is given by

$$\begin{aligned} \widetilde{\boldsymbol{\omega}}_t &= \boldsymbol{\omega}_t + \boldsymbol{b}_t^{\omega} + \boldsymbol{\eta}_t^{\omega} \\ \widetilde{\boldsymbol{a}}_t &= \boldsymbol{a}_t - \boldsymbol{R}_t{}^T \boldsymbol{g} + \boldsymbol{b}_t^{a} + \boldsymbol{\eta}_t^{a} \\ \dot{\boldsymbol{b}^{\omega}} &= \boldsymbol{\eta}^{b\omega}, \dot{\boldsymbol{b}^{a}} = \boldsymbol{\eta}^{ba} \end{aligned} \tag{2}$$

where g is the gravity vector, η^{ω}, η^{a}, $\eta^{b\omega}$, and η^{ba} are white Gaussian noise, and their standard deviations are σ_{η^a}, σ_{η^g}, $\sigma_{\eta^{ba}}$, and $\sigma_{\eta^{b\omega}}$, respectively.

4.1.1. Pose Prediction

Based on the posterior state $\hat{x}_{k-1}$ in the previous moment and the IMU measurements between (t_{k-1}, t_k), we can use the Euler integration to calculate the predicted poses. The recursive formula is

$$\begin{aligned} \boldsymbol{R}_{i+1} &= \boldsymbol{R}_i Exp((\widetilde{\boldsymbol{\omega}}_i - \boldsymbol{b}_i^{\omega} - \boldsymbol{\eta}_i^{\omega})\Delta t) \\ \boldsymbol{v}_{i+1} &= \boldsymbol{v}_i + \boldsymbol{g}\Delta t + \boldsymbol{R}_i(\widetilde{\boldsymbol{a}}_i - \boldsymbol{b}_i^{a} - \boldsymbol{\eta}_i^{a})\Delta t \\ \boldsymbol{p}_{i+1} &= \boldsymbol{p}_i + \boldsymbol{v}_i\Delta t + \frac{1}{2}\boldsymbol{g}\Delta t^2 + \frac{1}{2}\boldsymbol{R}_i(\widetilde{\boldsymbol{a}}_i - \boldsymbol{b}_i^{a} - \boldsymbol{\eta}_i^{a})\Delta t^2 \end{aligned} \tag{3}$$

where $Exp(\cdot)$ is the exponential map of manifold $SO(3)$.

These poses are sufficient to de-skew the point cloud and provide an initial value for registration. However, to help reject the outliers in Section 4.4, the covariance of these predicted poses also need to be estimated. The recursive formula of error state covariance propagation is

$$\boldsymbol{P}_{i+1} = \boldsymbol{F}_i \boldsymbol{P}_i {\boldsymbol{F}_i}^T + \boldsymbol{G}_i \boldsymbol{Q} {\boldsymbol{G}_i}^T \tag{4}$$

$$\boldsymbol{F}_i = \begin{bmatrix} Exp(-\boldsymbol{\omega}'\Delta t) & 0 & 0 & -\boldsymbol{J}_r(\boldsymbol{\omega}'\Delta t)\Delta t & 0 \\ 0 & \boldsymbol{I} & \boldsymbol{I}\Delta t & 0 & 0 \\ -\boldsymbol{R}_i[\boldsymbol{a}']_{\times}\Delta t & 0 & \boldsymbol{I} & 0 & -\boldsymbol{R}_i\Delta\, t \\ 0 & 0 & 0 & \boldsymbol{I} & 0 \\ 0 & 0 & 0 & 0 & \boldsymbol{I} \end{bmatrix}$$

$$\boldsymbol{G}_i = \begin{bmatrix} -\boldsymbol{J}_r(\boldsymbol{\omega}'\Delta t)\Delta t & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & -\boldsymbol{R}_i\Delta t & 0 & 0 \\ 0 & 0 & \boldsymbol{I}\Delta t & 0 \\ 0 & 0 & 0 & \boldsymbol{I}\Delta\, t \end{bmatrix}$$

where $\boldsymbol{\omega}' = \widetilde{\boldsymbol{\omega}}_i - \boldsymbol{b}_i^{\omega}$, $\boldsymbol{a}' = \widetilde{\boldsymbol{a}}_i - \boldsymbol{b}_i^{a}$, and $\boldsymbol{P}_i$ is the covariance matrix of system state at time t_i. The diagonal elements of the noise covariance matrix $\boldsymbol{Q}$ are ${\sigma_{\eta^{\omega}}}^2$, ${\sigma_{\eta^{a}}}^2$, ${\sigma_{\eta^{ba}}}^2$, and ${\sigma_{\eta^{b\omega}}}^2$. $\boldsymbol{J}_r$ is the right Jacobian of $SO(3)$. Note that the covariance of $\boldsymbol{R}$ is defined in the tangent space.

4.1.2. IMU Pre-Integration

The IMU pre-integration technique was first proposed in [32], and it has been widely applied in SLAM research. It uses IMU measurements between (t_{k-1}, t_k) to establish the constraint between two states x_{k-1} and x_k. The IMU pre-integration factor is calculated as follows:

$$\begin{aligned} \Delta\boldsymbol{R} &= {\boldsymbol{R}_{k-1}}^T \boldsymbol{R}_k = \prod_{t_i \in (t_{k-1}, t_k)} Exp((\widetilde{\boldsymbol{\omega}}_i - {\boldsymbol{b}^{\omega}}_i - {\boldsymbol{\eta}^{\omega}}_i)\Delta t) \\ \Delta\boldsymbol{v} &= \boldsymbol{R}_{k-1}^T(\boldsymbol{v}_k - \boldsymbol{v}_{k-1} - \boldsymbol{g}\Delta t) \\ &= \sum_{t_i \in (t_{k-1}, t_k)} \Delta\boldsymbol{R}_{k-1,i}(\widetilde{\boldsymbol{a}}_i - \boldsymbol{b}_i^{a} - \boldsymbol{\eta}_i^{a})\Delta t \\ \Delta\boldsymbol{p} &= {\boldsymbol{R}_{k-1}}^T \left(\boldsymbol{p}_k - \boldsymbol{p}_{k-1} - \boldsymbol{v}_{k-1}\Delta t - \frac{1}{2}\boldsymbol{g}\Delta t^2\right) \\ &= \sum_{t_i \in (t_{k-1}, t_k)} \left[\Delta\boldsymbol{v}_{k-1,i}\Delta t + \frac{1}{2}\Delta\boldsymbol{R}_{k-1,i}(\widetilde{\boldsymbol{a}}_i - \boldsymbol{b}_i^{a} - \boldsymbol{\eta}_i^{a})\Delta t^2\right] \end{aligned} \tag{5}$$

where ΔR, Δv, and Δp are the relative motion between two timestamps t_{k-1}, t_k. More details about the on-manifold IMU pre-integration can be found in [33].

4.2. Feature Extraction

The raw point cloud is measured in the LiDAR frame and distorted by the sensor's motion. Therefore, before feature extraction, it needs to be transformed to frame $\{FL\}$ and de-skewed. Suppose ${\boldsymbol{c}_i}^L$ is a point in raw point cloud scan $\mathcal{C} = \{\boldsymbol{c}_0, \boldsymbol{c}_1, \ldots, \boldsymbol{c}_n\}$:

$$ {\boldsymbol{c}_i}^{FL} = {\boldsymbol{T}_I^{FL}}^{-1} \boldsymbol{T}_{B_k}^{B_i} \boldsymbol{T}_I^{FL} {\boldsymbol{T}_M^{FL}}^{-1} \boldsymbol{R}_i^M \boldsymbol{T}_M^{FL} {\boldsymbol{c}_i}^L \tag{6} $$

where $\boldsymbol{R}_i^M$ is the rotation matrix generated by encoder data at time t_i, which represents the rotation of the LiDAR frame relative to the fixed LiDAR frame in the motor frame. $\boldsymbol{T}_{B_k}^{B_i} = {\boldsymbol{T}^{B_k}}^{-1} \boldsymbol{T}^{B_i}$, $\boldsymbol{T}^{B_i}$, and $\boldsymbol{T}^{B_k}$ are the poses of the body frame obtained by the linear interpolation of the predicted pose, and t_k is the timestamp of the latest point in the point cloud scan $\mathcal{C}$.

Our feature extraction method extracts planar features $\mathcal{F}_p$ and edge features $\mathcal{F}_e$ from the input point cloud as shown in Figure 4 . The workflow of the method is as follows:

(1) For a point $\boldsymbol{c}_i \in \mathcal{C}$, find its previous neighbors $\mathcal{N}_i^{pre} = \{\boldsymbol{c}_{i-k}, \ldots, \boldsymbol{c}_{i-1}\}$ and succeeding neighbors $\mathcal{N}_i^{succ} = \{\boldsymbol{c}_{i+1}, \ldots, \boldsymbol{c}_{i+k}\}$ in the same scan line.

(2) Calculate the features α, β of the point using

$$ \alpha = arccos\left(\frac{\boldsymbol{c}_i - \boldsymbol{c}_{i-1}}{\boldsymbol{c}_{i+1} - \boldsymbol{c}_i}\right) \tag{7} $$

$$ \beta = \frac{max(\|\boldsymbol{c}_{i+1} - \boldsymbol{c}_i\|, \|\boldsymbol{c}_i - \boldsymbol{c}_{i-1}\|)}{min(\|\boldsymbol{c}_{i+1} - \boldsymbol{c}_i\|, \|\boldsymbol{c}_i - \boldsymbol{c}_{i-1}\|)} \tag{8} $$

where α indicates the changing angle of the scan line at the point $\boldsymbol{c}_i$, which is used to characterize the smoothness. The points with $\alpha < \alpha^{thr}$ will be labeled as smooth points. β is the ratio of the distance from point $\boldsymbol{c}_i$ to $\boldsymbol{c}_{i-1}$ and $\boldsymbol{c}_{i+1}$, which is used to determine whether the point is an edge point.

(3) For point $\boldsymbol{c}_i$ with $\beta > \beta^{thr}$, if all points in its closer neighbors (depending on the closest point belonging to the neighbor $\mathcal{N}_i^{pre}$ or $\mathcal{N}_i^{succ}$) are smooth points, then add $\boldsymbol{c}_i$ to $\mathcal{F}_e$.

(4) For point $\boldsymbol{c}_i$ with $\beta \leq \beta^{thr}$, if all points in its previous and succeeding neighbors are smooth points, then add $\boldsymbol{c}_i$ to the candidate set of edge points.

(5) Use the standard LOAM-based method to extract planar features $\mathcal{F}_p$ and edge points $\mathcal{F}_e$, except that the edge points must belong to the candidate set.

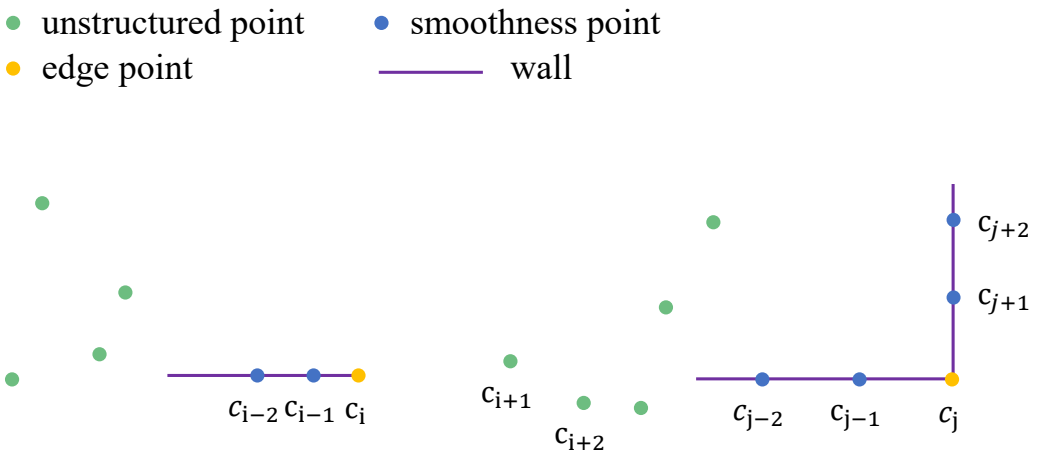


Figure 4. The illustration of the edge point extraction. As described in Steps 3 and 4, there are two types of edge points. The first type of edge point is at the end of the wall, and the second type of edge point is at the corners. These edge points can be identified from unstructured points by analyzing the properties of the points in their neighborhoods.

This modification was based on the consideration that there are many unstructured objects (e.g., vegetation) in outdoor environments, which will degrade the performance of edge extraction. A comparison result is shown in Figure 5. In our experiment, α^{thr} was set to 15° and k and β^{thr} were set to 2 and 4, respectively, according to [34].

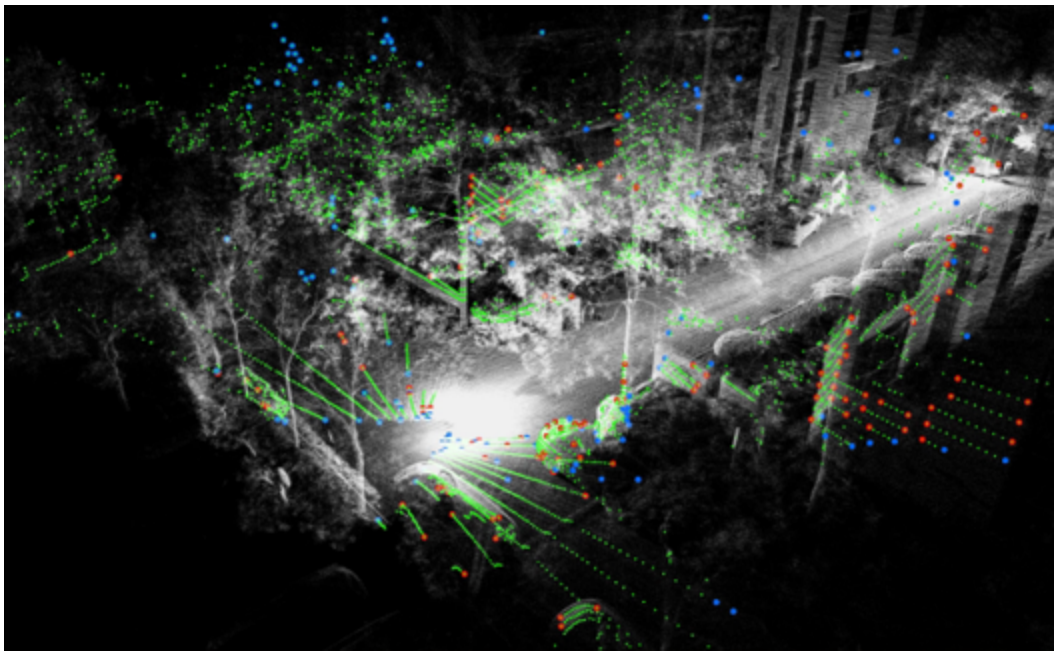


Figure 5. Feature point extraction result. The green points are the raw point cloud. The edge points extracted by the standard LOAM-based extraction method are shown in blue, and the edge points extracted by our method are shown in red. It can be seen that many blue points are located in the vegetation, and these noises will affect the accuracy of the alignment.

4.3. Scan-to-Map Registration

Similar to [19], for an input point cloud scan C, we used the pose predicted by the IMU to find the adjacent point cloud scans in the global map and merged them into the feature map $\mathcal{M}_p$ and $\mathcal{M}_e$ for data association. The map is downsampled by a voxel filter to accelerate the computation. Then, we find the k nearest neighboring points N_f of each feature point $f_i \in \mathcal{F}_p$ or $\mathcal{F}_e$ in the corresponding feature map and denote its nearest point as m_i. For $f_i \in \mathcal{F}_p$; the plane normal vector $\boldsymbol{n}_p$ of its $\mathcal{N}_f$ is computed, and for $f_i \in \mathcal{F}_e$, the line direction $\boldsymbol{n}_e$ of its N_f is computed. In addition, as the LiDAR is continuously rotating, to ensure the reliability of the matching in the initial stage, the initial global map is constructed by keeping the sensor platform stationary for about 3 s.

According to the results of feature matching, the LiDAR residual r_{LiDAR} can be computed using the point-to-plane d_p and point-to-line distance d_e.

$$d_p = \boldsymbol{n}_p{}^T(\boldsymbol{R}\boldsymbol{f}_i + \boldsymbol{p} - \boldsymbol{m}_i) \tag{9}$$

$$d_e = \|\boldsymbol{n}_e \times (\boldsymbol{R}\boldsymbol{f}_i + \boldsymbol{p} - \boldsymbol{m}_i)\| \tag{10}$$

where $\boldsymbol{R}$ and $\boldsymbol{p}$ are the predicted orientation and position, $\boldsymbol{n}_p$ and $\boldsymbol{n}_e$ are normalized, and $\boldsymbol{m}_i$ is the corresponding point of feature point $\boldsymbol{f}_i$ in the global map.

4.4. Adaptive Scan Accumulation

The spatial distribution of the point cloud has an important influence on the accuracy of the registration result. Taking point-to-plane registration as an example, the matched points should involve at least three non-parallel planes. Unlike the point cloud obtained by the fixed mounted LiDAR, the LiDAR in our device is continuously rotating, resulting in

the continuous change of the spatial distribution of the obtained point cloud. As shown in Figure 6, using only a single frame of the point cloud sometimes fails to provide reliable registration results. A simple solution is to accumulate multi-scan point clouds before each registration, but this will reduce the computational efficiency, so we propose a method to adaptively decide whether to perform point cloud accumulation and how many scans need to be accumulated according to the spatial distribution of the point cloud.

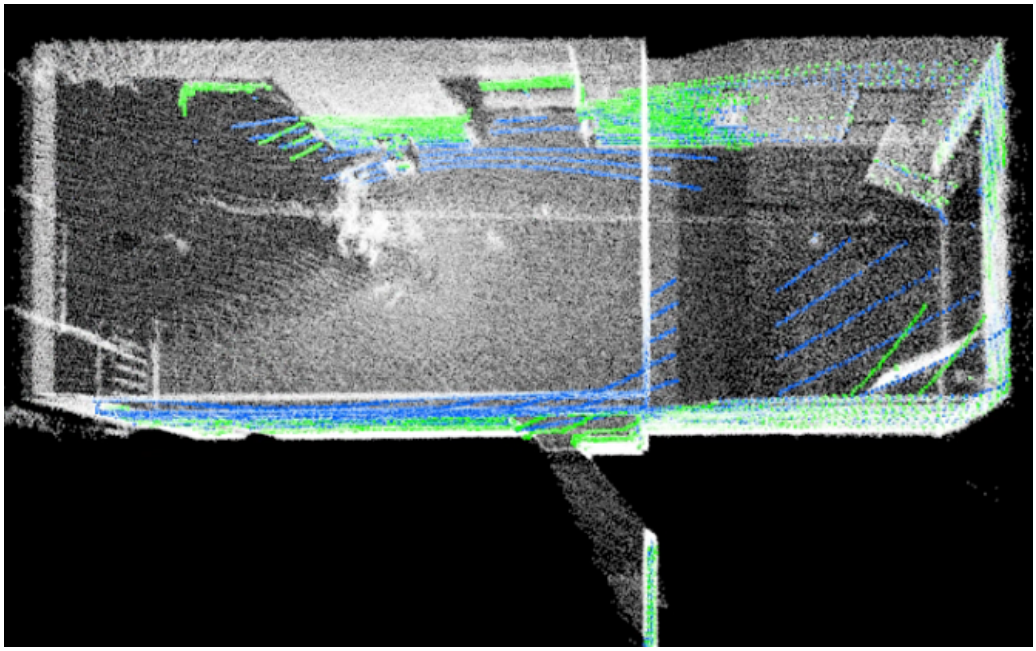


Figure 6. Illustration of multi-scan point clouds' accumulation. The first scan point cloud is rendered in green and only contains a few points in the horizontal plane, which will lead to an inaccurate registration result, especially in the Z direction. This problem can be solved by merging it with the next scan point cloud.

4.4.1. Features' Distribution Inspection

To ensure the registration results are reliable, we used the covariance of the registration result as the indicator. In the least-squares registration problem, the normal equation formed by matched feature points is

$$\boldsymbol{J}^T\boldsymbol{J}\boldsymbol{x} = \boldsymbol{J}^T\boldsymbol{b} \tag{11}$$

where $\boldsymbol{J} \in \mathbb{R}^{n\times 6}$ is constructed by stacking Jacobian matrices $\boldsymbol{J}_p^{d_p}$ and $\boldsymbol{J}_p^{d_e}$, n is the number of matched points, and $b \in \mathbb{R}^{n\times 1}$ is constructed by stacking distance residuals d_p and d_e. The solution to Equation (11) is given by

$$\hat{\boldsymbol{x}} = \left(\boldsymbol{J}^T\boldsymbol{J}\right)^{-1}\boldsymbol{J}^T\boldsymbol{b} \tag{12}$$

The covariance of the estimated pose is

$$E\left\{\hat{\boldsymbol{x}}^T\hat{\boldsymbol{x}}\right\} = \left(\boldsymbol{J}^T\boldsymbol{J}\right)^{-1}\boldsymbol{J}^T E\left\{\boldsymbol{b}^T\boldsymbol{b}\right\}\boldsymbol{J}\left(\boldsymbol{J}^T\boldsymbol{J}\right)^{-1} \tag{13}$$

where $E\{\cdot\}$ is the expected value operator. The covariance of $\boldsymbol{b}$ is determined by the laser measurements, and each laser observation can be treated as an independent observation. Therefore, its covariance is:

$$E\left\{\boldsymbol{b}^T\boldsymbol{b}\right\} = \sigma_l^2\boldsymbol{I} \tag{14}$$

where σ_l is the accuracy of the laser measurement, which can be set depending on the model of the LiDAR. Substituting Equation (14) into Equation (13), then

$$E\left\{\hat{\boldsymbol{x}}^T\hat{\boldsymbol{x}}\right\} = \left(\boldsymbol{J}^T\boldsymbol{J}\right)^{-1}\sigma_l^2 = \boldsymbol{H}\sigma_l^2 \tag{15}$$

Equation (15) reveals how the spatial distribution of matched points in matrix $\boldsymbol{J}$ affects the covariance of the registration result. If the spatial distribution is suitable for registration, the covariance of the estimated pose should not vary greatly in all directions. Suppose $\boldsymbol{H}_{tran} \in \mathbb{R}^{3\times3}$ is the submatrix of $\boldsymbol{H}$ corresponding to the translation part in the pose; the ratio γ can be computed by

$$\gamma = \frac{\sqrt{max(diag(\boldsymbol{H}_{tran}))}}{\sqrt{min(diag(\boldsymbol{H}_{tran}))}} \tag{16}$$

where operator $diag(\cdot)$ extracts the diagonal elements of the matrix as a vector. The smaller the value of γ, the more uniform the distribution of the point cloud is, and vice versa. Here, we set the threshold $\gamma_{th} = 3$. If $\gamma < \gamma_{th}$, the matched pairs will be accepted, and these plane and edge features will be added to the factor graph as constraints; otherwise, the extracted features will be accumulated to the next scan.

4.4.2. Outlier Removal in Matched Features

Since the point cloud in a single scan is sparse, there will inevitably be errors in the feature extraction results. These incorrectly matched point pairs will interfere with the features' distribution inspection and reduce the accuracy of pose estimation. To remove them while performing distribution inspection, an outlier removal algorithm based on the propagation of covariance is applied. We assumed that the error in the global map can be ignored, which means that the distance residual is mainly caused by the error of the predicted pose and the error of the laser measurement. Then, the standard deviation of the distance residuals can be computed by

$$\sigma_{d_p} = \sqrt{\boldsymbol{J}_p^{d_p}\,\boldsymbol{D}_p\,\boldsymbol{J}_p^{d_p\,T} + \boldsymbol{J}_l^{d_p}\,\boldsymbol{D}_l\,\boldsymbol{J}_l^{d_p\,T}} \tag{17}$$

$$\sigma_{d_e} = \sqrt{\boldsymbol{J}_p^{d_e}\boldsymbol{D}_p\boldsymbol{J}_p^{d_e\,T} + \boldsymbol{J}_l^{d_e}\boldsymbol{D}_l\boldsymbol{J}_l^{d_e\,T}} \tag{18}$$

$$\boldsymbol{J}_p^{d_p} = \begin{bmatrix}\frac{\partial d_p}{\partial \boldsymbol{R}} & \frac{\partial d_p}{\partial \boldsymbol{p}}\end{bmatrix} = \begin{bmatrix}\boldsymbol{n}_p{}^T\boldsymbol{R}\lfloor-\boldsymbol{f}_i\rfloor_\times & \boldsymbol{n}_p{}^T\end{bmatrix},$$

$$\boldsymbol{J}_l^{d_p} = \boldsymbol{n}_p{}^T\boldsymbol{R}$$

$$\boldsymbol{J}_p^{d_e} = \begin{bmatrix}\frac{\partial d_e}{\partial \boldsymbol{R}} & \frac{\partial d_e}{\partial \boldsymbol{p}}\end{bmatrix} = \begin{bmatrix}\frac{\boldsymbol{v}_{d_e}{}^T}{\|\boldsymbol{v}_{d_e}\|}\lfloor\boldsymbol{n}_e\rfloor_\times\boldsymbol{R}\lfloor-\boldsymbol{f}_i\rfloor_\times & \frac{\boldsymbol{v}_{d_e}{}^T}{\|\boldsymbol{v}_{d_e}\|}\lfloor\boldsymbol{n}_e\rfloor_\times\end{bmatrix},$$

$$\boldsymbol{J}_l^{d_e} = \frac{\boldsymbol{v}_{d_e}{}^T}{\|\boldsymbol{v}_{d_e}\|}\lfloor\boldsymbol{n}_e\rfloor_\times\boldsymbol{R}, \quad \boldsymbol{v}_{d_e} = \boldsymbol{n}_e \times (\boldsymbol{R}\boldsymbol{f}_i + \boldsymbol{p} - \boldsymbol{m}_i)$$

where D_p is the covariance of the predicted pose and can be obtained by Equation (4) and D_l is the accuracy of the LiDAR point.

The point pair with $d_p \geq 3\sigma_{d_p}$ or $d_e \geq 3\sigma_{d_e}$ is considered an outlier and removed. In theory, if the distance residuals follow a Gaussian distribution, we can remove most of the mismatched point pairs while retaining 99.5% of the correct matching results.

4.5. Loop Closure Detection

Our loop closure detection method was developed based on Cartographer [35]. As shown in Algorithm 1, there are two main improvements compared to the original algorithm: (1) point-to-plane ICP is used to replace the original probability occupancy-grid-

based registration method in the fine registration stage; (2) the loop closure detection result is checked using the previous scan. Due to the rotation of the LiDAR, there is a noticeable difference even between two adjacent scans. This can provide auxiliary information to help reject false detection results. If the loop closure detection result of the current scan is correct, then the registration results $\boldsymbol{T}_{S_l}^{C_k}$ and $\boldsymbol{T}_{S_l}^{C_k{}^{-1}}$ should be consistent with the odometry transformation $\boldsymbol{T}_{C_k}^{C_{k-1}}$. The threshold d_{th} was set to 5 cm in the experiment.

Algorithm 1: Loop closure detection.

Input: scan $\mathcal{C}_k$, submap $\mathcal{S}_l$
Output: loop closure residuals $\mathcal{R}_{loop}$
`// Obtain coarse registration result and score`
$\boldsymbol{T}_{S_l}^{C_k}, s \leftarrow$ *Branch-and-boundScanMatch*$(\mathcal{C}_k, \mathcal{S}_l)$;
if $s > s_{th}$ **then**
  `// Refine coarse registration result`
  $\boldsymbol{T}_{S_l}^{C_k} \leftarrow$ *Point-to-planeICP*$(\mathcal{C}_k, \mathcal{S}_l, \boldsymbol{T}_{S_l}^{C_k})$;
  `// Check ICP result using previous scan`
  $\boldsymbol{T}_{S_l}^{C_{k-1}} \leftarrow$ *Point-to-planeICP*$(\mathcal{C}_{k-1}, \mathcal{S}_l, \boldsymbol{T}_{C_k}^{C_{k-1}} \boldsymbol{T}_{S_l}^{C_k})$;
  $d \leftarrow$*TranslationError*$(\boldsymbol{T}_{C_k}^{C_{k-1}}, \boldsymbol{T}_{S_l}^{C_k{}^{-1}} \boldsymbol{T}_{S_l}^{C_{k-1}})$;
  if $d < d_{th}$ **then**
    Add residual $r(\mathcal{C}_k, \mathcal{S}_l, \boldsymbol{T}_{S_l}^{C_k})$ to $\mathcal{R}_{loop}$;
    Add residual $r(\mathcal{C}_{k-1}, \mathcal{S}_l, \boldsymbol{T}_{S_l}^{C_{k-1}})$ to $\mathcal{R}_{loop}$;
  end
end

5. Experiments

Since there is no publicly available dataset containing spinning actuated LiDAR and IMU data, we evaluated the performance of the Spin-LOAM algorithm using the data collected in indoor and outdoor environments with the device shown in Figure 7. In the experiments, the sampling frequencies of the Velodyne VLP-16 LiDAR and Sensonor STIM300 IMU were 10 Hz and 200 Hz, respectively. The angular velocity of the step motor was set to 4.5 rad/s, and the angular resolution of the encoder was 10 bit. All experiments were conducted on an Intel NUC computer with an Intel Core i5-1135G7 CPU and 16 GB memory.

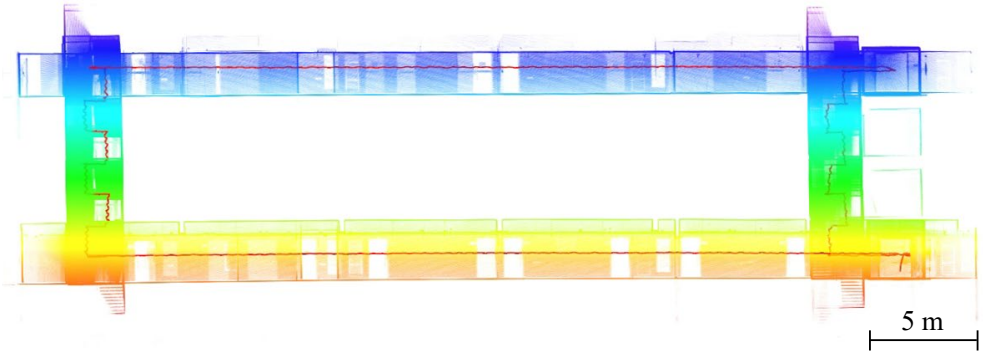
(**a**) (**b**)

Figure 7. Overview of the experimental sensor platform. (**a**) The sensor platform is carried by a backpack. (**b**) The sensor platform is carried by a UAV.

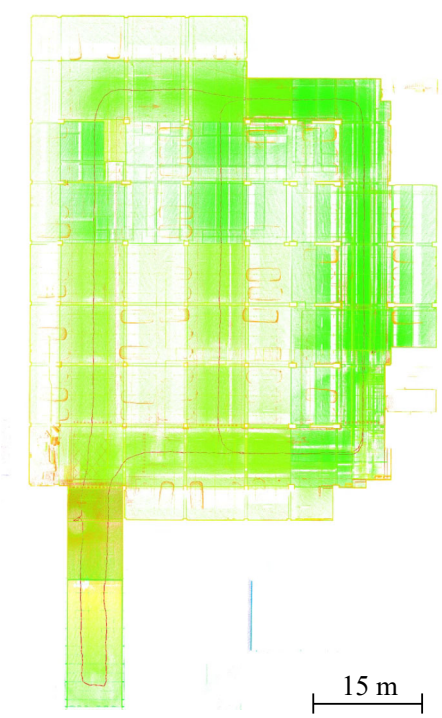
5.1. Evaluation in Indoor Environments

To verify the accuracy and effectiveness of Spin-LOAM, we first conducted tests in indoor environments. We compared our algorithm with two state-of-the-art LIO algorithms, FAST-LIO2 and LIO-SAM (modified for compatibility with a 6-axis IMU). In the indoor tests, the map voxel size in all algorithms was set to 0.2 m. The standard deviations of the noise related to the IMU and LiDAR were uniformly set according to the device model.

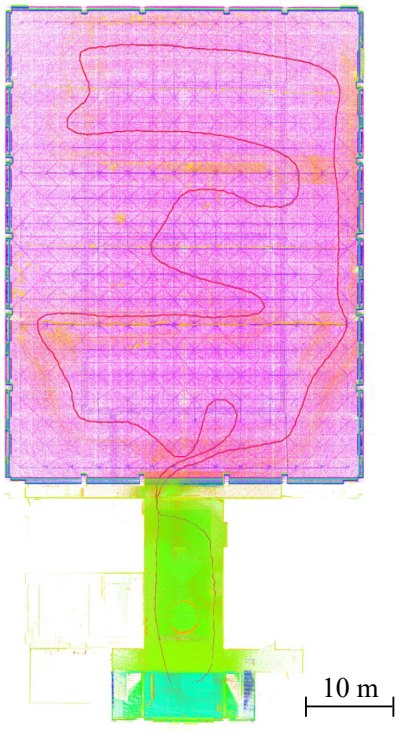
As shown in Figure 8, we collected three datasets in different environments using a backpack to evaluate the performance of our algorithm. Data 1 is two narrow corridors on different floors connected by stairs; Data 2 is an underground garage; Data 3 is a badminton hall. Since it was difficult to obtain the ground truth trajectory in the indoor environment, we returned to the starting position at the end of the trajectory and used the end-to-end translation error as the metric. The qualitative analysis results are given in Table 1, where "Spin-LOAM (odom)" represents our algorithm without loop closure and "Spin-LOAM (full)" represents the full SLAM algorithm.



(a) Data 1



(b) Data 2



(c) Data 3

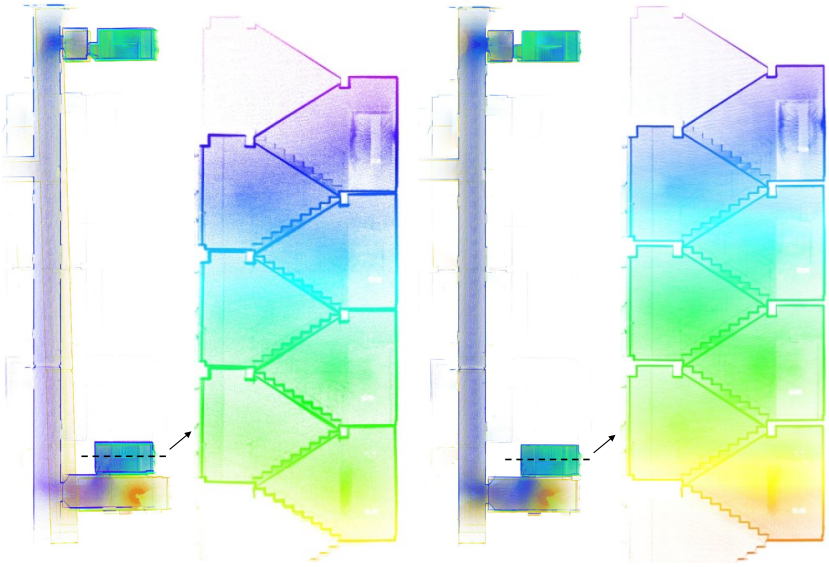
Figure 8. Mapping results of our algorithm in indoor environments. The trajectories are represented by the red lines in the figure.

Table 1. End-to-end translation error (m).

	Length (m)	FAST-LIO2	LIO-SAM	Spin-LOAM (Odom)	Spin-LOAM (Full)
Data 1	189.0993	2.2631	/	0.6976	**0.0090**
Data 2	382.3672	0.0210	0.0128	0.0126	**0.0114**
Data 3	315.1776	0.0166	**0.0145**	0.0175	0.0164

Bold font represents better results.

Data 1 is the most-challenging scene among the three indoor datasets. In this scene, LIO-SAM failed to finish the localization, and FAST-LIO2 also suffered from severe drift. Our algorithm achieved the lowest odometry drift and successfully corrected the drift through loop closure detection, as shown in Figure 9. In Data 2, our algorithm and LIO-SAM achieved similar localization accuracy, and FAST-LIO2 was slightly worse because it has no loop closure detection. However, the accuracy of our pure odometry method was comparable to LIO-SAM, which validates the effectiveness of our algorithm. In Data 3, the accuracy of all algorithms was at the same level, because the scene was free of occlusions and full of plane features. It is worth noting that the height of the roof in this scene was about 15 m, but our device completely acquired the point cloud of the entire scene with only one LiDAR. This reveals the advantage of the spinning actuated LiDAR system.

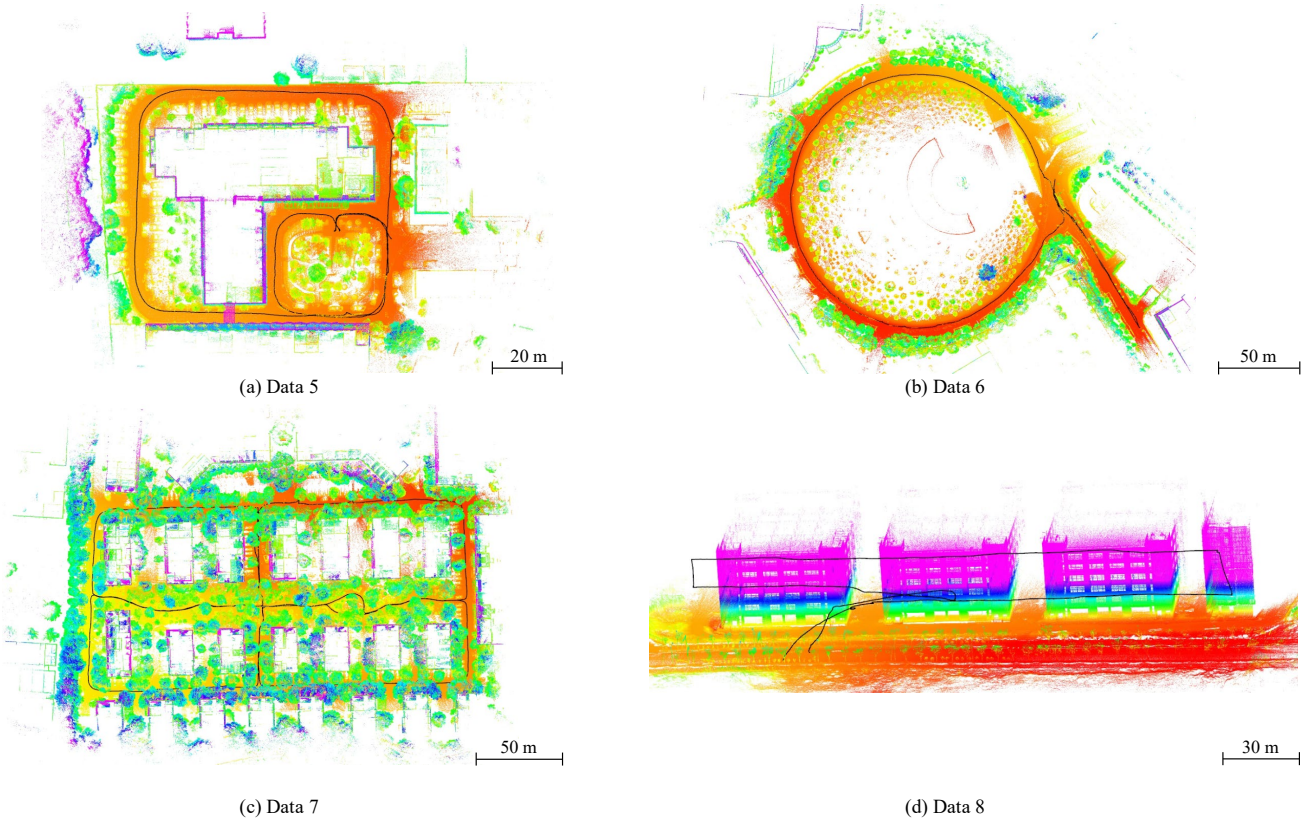


(a) FAST-LIO2 (b) Spin-LOAM

Figure 9. Comparison of the mapping results of the stairs in Data 1.

5.2. Evaluation in Outdoor Environments

In the outdoor test, the map voxel size in Spin-LOAM was changed to 0.4 m, and the voxel size in FAST-LIO2 and LIO-SAM was set according to the default value. Figure 10 gives an overview of the datasets collected in the outdoor environments. Data 5 was collected around a building; Data 6 was collected on a large ring road. Data 7 was collected in a residential area. Data 8 was collected using a drone on a construction site. To quantitatively compare the performance of the algorithms, we used the GNSS RTK trajectories as the ground truth and computed the absolute trajectory error (ATE) of the trajectories.



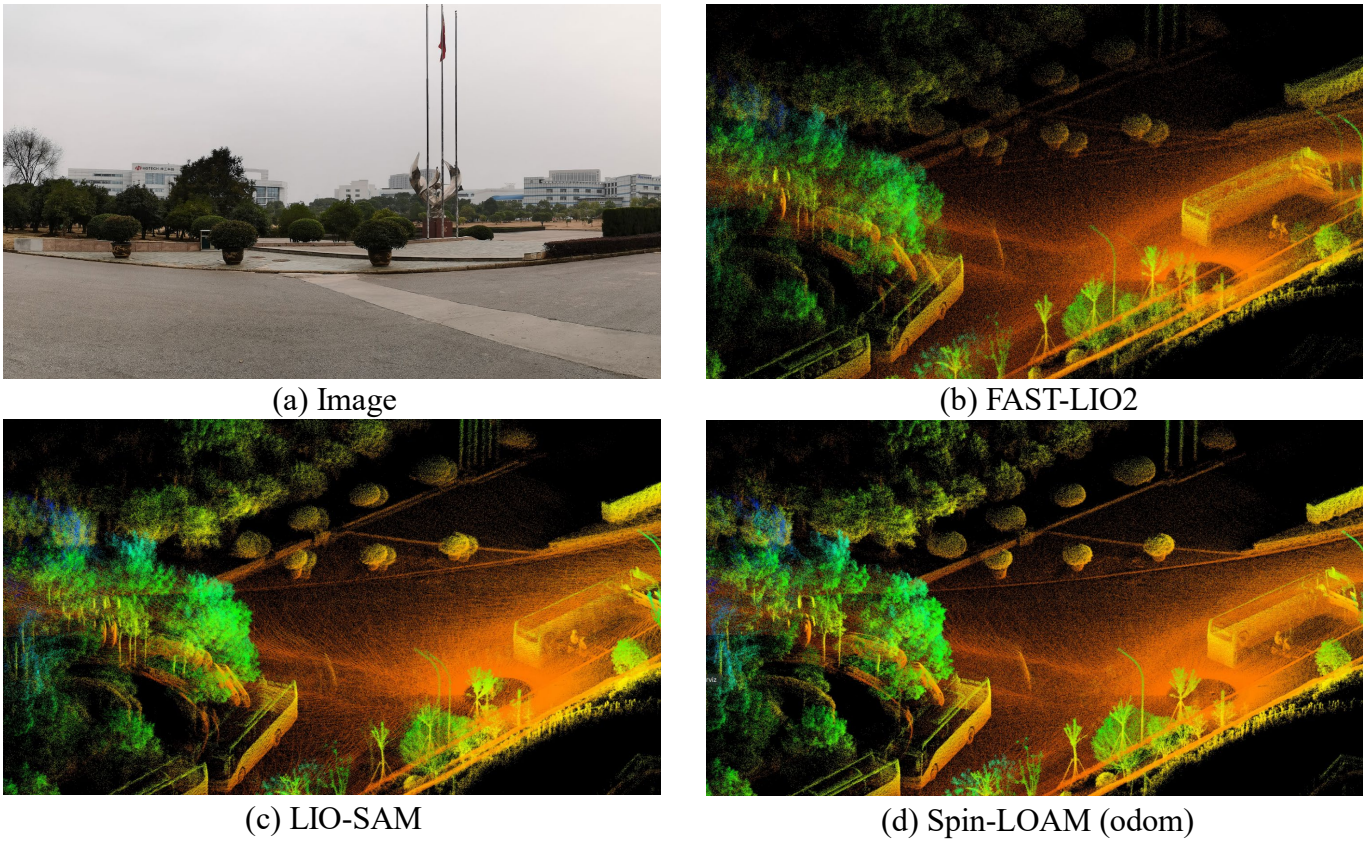
(a) Data 5 (b) Data 6

(c) Data 7 (d) Data 8

Figure 10. Mapping results of our algorithm in outdoor environments. The ground truth trajectories generated by GNSS are represented by the black lines.

Table 2 shows that our algorithm achieved the best accuracy in the outdoor environments. FAST-LIO2 also performed very well in the experiments with only significant drift in Data 6 and completed all tests as our algorithm. However, LIO-SAM failed to finish the localization in Data 6 and Data 8. The reason for its failure in Data 6 is that it directly uses the ICP algorithm for loop closure. This strategy is not suitable when large drift occurs, and the accumulated errors in the large ring road make the LIO-SAM algorithm fail. After turning off the loop closure, the accuracy of LIO-SAM in Data 6 was 0.4908 m. Figure 11 gives a detailed comparison of the point cloud at the road junction. It can be seen that, even without the loop closure, our algorithm still maintained high accuracy after walking through a long loop. In Data 8, the drone performed an aggressive motion to test the robustness of the algorithm, in which the maximum angular velocity was over 780°/s and the maximum linear velocity was over 6.5 m/s. This test proved that our tightly coupled algorithm can work when aggressive motion occurs.

An ablation study was conducted to further analyze the contribution of our proposed adaptive scan accumulation method. "Spin-LOAM (wo-asa)" in Table 2 represents our algorithm without adaptive scan accumulation and loop closure. Comparing it with "Spin-LOAM (odom)", the results showed that our method can improve the accuracy, especially in complex environments such as Data 6 and 7. This is because the FOV of the LiDAR is limited; it is often occluded by trees or can only scan the ground in these environments, which will lead to an uneven distribution of features in a single scan. Our method can alleviate this problem during scan-to-map registration, which can help to improve the accuracy and robustness of registration.



(a) Image (b) FAST-LIO2

(c) LIO-SAM (d) Spin-LOAM (odom)

Figure 11. Point cloud maps generated by different algorithms in Data 6.

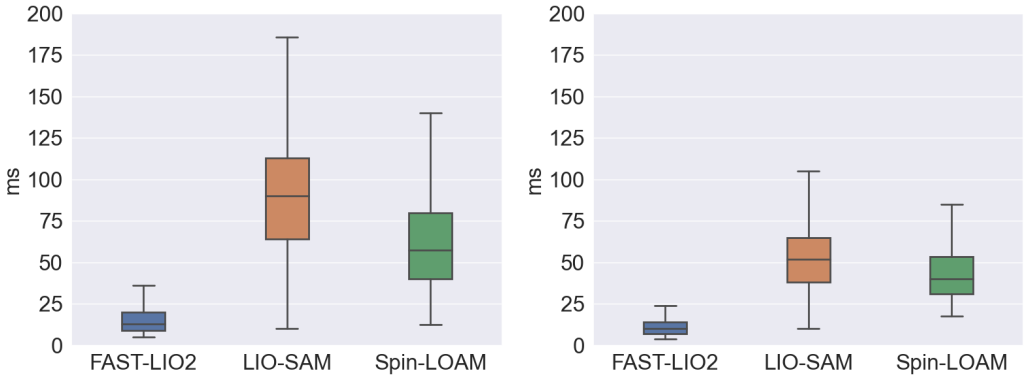
Table 2. Absolute trajectory error (ATE) of trajectories (m).

	Length (m)	FAST-LIO2	LIO-SAM	Spin-LOAM (wo-asa)	Spin-LOAM (Odom)	Spin-LOAM (Full)
Data 5	415.3925	0.0799	0.0854	0.0760	0.0753	**0.0725**
Data 6	782.0214	0.8844	(0.4908)	0.2623	0.2323	**0.2175**
Data 7	1335.2931	0.2271	0.2771	0.2064	0.1871	**0.1786**
Data 8	623.8667	0.1670	/	0.1431	0.1353	**0.1308**

Bold font represents better results.

5.3. Runtime Analysis

We selected indoor data and an outdoor data, respectively, for the algorithm performance evaluation. The processing time per scan of each algorithm is shown in Figure 12. FAST-LIO2 was much faster than LIO-SAM and our algorithm because it is a filter-based algorithm and does not require feature extraction. The average time cost per scan of our algorithm was 63.2 ms indoors and 43.8 ms outdoors. The computation time was more for the indoor data because the map voxel size had a significant impact on the performance. The average time cost for the loop closure was 808.4 ms indoors and 328.7 ms outdoors. Since the loop closure was performed by a separate thread in the background, it did not affect the real-time performance of our algorithm.



(a) Indoor (b) Outdoor

Figure 12. The evaluation results of the time cost per scan for each algorithm. The selected indoor data and outdoor data are Data 2 and Data 7, respectively, as they have the longest trajectories.

6. Conclusions

In this paper, we presented a tightly coupled laser–inertial SLAM algorithm specifically designed for a spinning actuated LiDAR system. In the front-end, to mitigate the influence of the unstable spatial distribution of the point cloud caused by the continuously rotating LiDAR, an adaptive scan accumulation method based on point cloud distribution inspection was adopted. In the back-end, a voxel-grid-based loop closure detection method was used to reduce the drift. We use the previous scan point cloud to assist in eliminating errors in the loop closure detection results. The experimental results demonstrated that our algorithm achieves high-precision localization results in various complex indoor and outdoor environments. We are committed to further refining and improving our algorithm, with a focus on improving its robustness in more extreme environmental conditions. One potential avenue for improvement is the integration of semantic information from the point cloud, which will aid in loop closure detection and removal of dynamic objects.

Author Contributions: Methodology, J.D.; Investigation, Y.Z.; Resources, C.C.; Writing—original draft, J.D.; Supervision, L.Y.; Funding acquisition, L.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Science and Technology Major Project of Hubei Province under Grant 2021AAA010.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Li, Y.; Ibanez-Guzman, J. LiDAR for Autonomous Driving: The Principles, Challenges, and Trends for Automotive LiDAR and Perception Systems. *IEEE Signal Process. Mag.* **2020**, *37*, 50–61. [CrossRef]
2. Bolourian, N.; Hammad, A. LiDAR-equipped UAV path planning considering potential locations of defects for bridge inspection. *Autom. Constr.* **2020**, *117*, 103250. [CrossRef]
3. Beland, M.; Parker, G.; Sparrow, B.; Harding, D.; Chasmer, L.; Phinn, S.; Antonarakis, A.; Strahler, A. On promoting the use of LiDAR systems in forest ecosystem research. *For. Ecol. Manag.* **2019**, *450*, 117484. [CrossRef]
4. Raj, T.; Hanim Hashim, F.; Baseri Huddin, A.; Ibrahim, M.F.; Hussain, A. A survey on LiDAR scanning mechanisms. *Electronics* **2020**, *9*, 741. [CrossRef]
5. Li, K.; Li, M.; Hanebeck, U.D. Towards high-performance solid-state-LiDAR-inertial odometry and mapping. *IEEE Robot. Autom. Lett.* **2021**, *6*, 5167–5174. [CrossRef]
6. Alsadik, B.; Remondino, F. Flight planning for LiDAR-based UAS mapping applications. *ISPRS Int. J. Geo-Inf.* **2020**, *9*, 378. [CrossRef]
7. Liu, X.; Zhang, F.Z. Extrinsic Calibration of Multiple LiDARs of Small FoV in Targetless Environments. *IEEE Robot. Autom. Lett.* **2021**, *6*, 2036–2043. [CrossRef]
8. Nguyen, T.M.; Yuan, S.; Cao, M.; Lyu, Y.; Nguyen, T.H.; Xie, L. MILIOM: Tightly Coupled Multi-Input LiDAR-Inertia Odometry and Mapping. *IEEE Robot. Autom. Lett.* **2021**, *6*, 5573–5580. [CrossRef]

9. Jiao, J.; Ye, H.; Zhu, Y.; Liu, M. Robust Odometry and Mapping for Multi-LiDAR Systems With Online Extrinsic Calibration. *IEEE Trans. Robot.* **2022**, *38*, 351–371. [CrossRef]
10. Zhang, D.; Gong, Z.; Chen, Y.; Zelek, J.; Li, J. SLAM-based multi-sensor backpack LiDAR systems in gnss-denied environments. In Proceedings of the IGARSS 2019 IEEE International Geoscience and Remote Sensing Symposium, Yokohama, Japan, 28 July–2 August 2019; pp. 8984–8987.
11. Velas, M.; Spanel, M.; Sleziak, T.; Habrovec, J.; Herout, A. Indoor and outdoor backpack mapping with calibrated pair of velodyne LiDARs. *Sensors* **2019**, *19*, 3944. [CrossRef]
12. Zhang, J.; Singh, S. LOAM: LiDAR Odometry and Mapping in Real-time. *Robot. Sci. Syst.* **2014**, *2*, 9.
13. Shan, T.; Englot, B. Lego-loam: Lightweight and ground-optimized LiDAR odometry and mapping on variable terrain. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 4758–4765.
14. Zhou, P.; Guo, X.; Pei, X.; Chen, C. T-LOAM: Truncated Least Squares LiDAR-Only Odometry and Mapping in Real Time. *IEEE Trans. Geosci. Remote. Sens.* **2021**, *60*, 5701013. [CrossRef]
15. Liu, Z.; Zhang, F. BALM: Bundle Adjustment for LiDAR Mapping. *IEEE Robot. Autom. Lett.* **2021**, *6*, 3184–3191. [CrossRef]
16. Zhou, L.; Koppel, D.; Kaess, M. LiDAR SLAM With Plane Adjustment for Indoor Environment. *IEEE Robot. Autom. Lett.* **2021**, *6*, 7073–7080. [CrossRef]
17. Zhou, L.; Wang, S.; Kaess, M. π-LSAM: LiDAR smoothing and mapping with planes. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA), Xi'an, China, 30 May–5 June 2021; pp. 5751–5757.
18. Chen, X.; Milioto, A.; Palazzolo, E.; Giguere, P.; Behley, J.; Stachniss, C. Suma++: Efficient LiDAR-based semantic slam. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 3–8 November 2019; pp. 4530–4537.
19. Shan, T.; Englot, B.; Meyers, D.; Wang, W.; Ratti, C.; Rus, D. Lio-sam: Tightly-coupled LiDAR inertial odometry via smoothing and mapping. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 25–29 October 2020; pp. 5135–5142.
20. Ye, H.; Chen, Y.; Liu, M. Tightly coupled 3D LiDAR inertial odometry and mapping. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 3144–3150.
21. Xu, W.; Zhang, F. Fast-lio: A fast, robust LiDAR-inertial odometry package by tightly-coupled iterated kalman filter. *IEEE Robot. Autom. Lett.* **2021**, *6*, 3317–3324. [CrossRef]
22. Xu, W.; Cai, Y.; He, D.; Lin, J.; Zhang, F. FAST-LIO2: Fast Direct LiDAR-Inertial Odometry. *IEEE Trans. Robot.* **2022**, *38*, 2053–2073. [CrossRef]
23. Lv, J.; Hu, K.; Xu, J.; Liu, Y.; Ma, X.; Zuo, X. Clins: Continuous-time trajectory estimation for LiDAR-inertial system. In Proceedings of the 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Prague, Czech Republic, 27 September–1 October 2021; pp. 6657–6663.
24. Bosse, M.; Zlot, R.; Flick, P. Zebedee: Design of a Spring-Mounted 3-D Range Sensor with Application to Mobile Mapping. *IEEE Trans. Robot.* **2012**, *28*, 1104–1119. [CrossRef]
25. Kaul, L.; Zlot, R.; Bosse, M. Continuous-Time Three-Dimensional Mapping for Micro Aerial Vehicles with a Passively Actuated Rotating Laser Scanner. *J. Field Robot.* **2016**, *33*, 103–132. [CrossRef]
26. Park, C.; Moghadam, P.; Kim, S.; Elfes, A.; Fookes, C.; Sridharan, S. Elastic LiDAR Fusion: Dense Map-Centric Continuous-Time SLAM. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; pp. 1206–1213. [CrossRef]
27. Fang, Z.; Zhao, S.; Wen, S. A Real-time and Low-cost 3D SLAM System Based on a Continuously Rotating 2D Laser Scanner. In Proceedings of the 2017 IEEE 7th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER), Honolulu, HI, USA, 31 July–4 August 2017; pp. 454–459. [CrossRef]
28. Zhen, W.; Hu, Y.; Liu, J.; Scherer, S. A Joint Optimization Approach of LiDAR-Camera Fusion for Accurate Dense 3-D Reconstructions. *IEEE Robot. Autom. Lett.* **2019**, *4*, 3585–3592. [CrossRef]
29. Oelsch, M.; Karimi, M.; Steinbach, E. R-LOAM: Improving LiDAR Odometry and Mapping With Point-to-Mesh Features of a Known 3D Reference Object. *IEEE Robot. Autom. Lett.* **2021**, *6*, 2068–2075. [CrossRef]
30. Karimi, M.; Oelsch, M.; Stengel, O.; Babaians, E.; Steinbach, E. LoLa-SLAM: Low-latency LiDAR SLAM using Continuous Scan Slicing. *IEEE Robot. Autom. Lett.* **2021**, 2248–2255. [CrossRef]
31. Benson, M.; Nikolaidis, J.; Clayton, G.M. Lissajous-like scan pattern for a nodding multi-beam LiDAR. In Proceedings of the Dynamic Systems and Control Conference, Atlanta, GA, USA, 30 September–3 October 2018; Volume 51906, p. V002T24A007.
32. Lupton, T.; Sukkarieh, S. Efficient integration of inertial observations into visual SLAM without initialization. In Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, St. Louis, MO, USA, 10–15 October 2009; pp. 1547–1552.
33. Forster, C.; Carlone, L.; Dellaert, F.; Scaramuzza, D. On-Manifold Preintegration for Real-Time Visual–Inertial Odometry. *IEEE Trans. Robot.* **2017**, *33*, 1–21. [CrossRef]

34. Chen, P.; Shi, W.; Bao, S.; Wang, M.; Fan, W.; Xiang, H. Low-Drift Odometry, Mapping and Ground Segmentation Using a Backpack LiDAR System. *IEEE Robot. Autom. Lett.* **2021**, *6*, 7285–7292. [CrossRef]
35. Hess, W.; Kohler, D.; Rapp, H.; Andor, D. Real-time loop closure in 2D LiDAR SLAM. In Proceedings of the IEEE International Conference on Robotics and Automation, Stockholm, Sweden, 16–21 May 2016.