sensors

Article

A Simultaneous Localization and Mapping System Using the Iterative Error State Kalman Filter Judgment Algorithm for Global Navigation Satellite System

Bo You [1,2], Guangjin Zhong [1], Chen Chen [1,2,*], Jiayu Li [1,2] and Ersi Ma [1]

[1] Heilongjiang Provincial Key Laboratory of Complex Intelligent System and Integration, Harbin University of Science and Technology, Harbin 150080, China; youbo@hrbust.edu.cn (B.Y.); 2120510126@stu.hrbust.edu.cn (G.Z.); lijiayu@hrbust.edu.cn (J.L.); maersi97@163.com (E.M.)

[2] Key Laboratory of Intelligent Technology for Cutting and Manufacturing Ministry of Education, Harbin University of Science and Technology, Harbin 150080, China

* Correspondence: danny_cc@hrbust.edu.cn

Abstract: Outdoor autonomous mobile robots heavily rely on GPS data for localization. However, GPS data can be erroneous and signals can be interrupted in highly urbanized areas or areas with incomplete satellite coverage, leading to localization deviations. In this paper, we propose a SLAM (Simultaneous Localization and Mapping) system that combines the IESKF (Iterated Extended Kalman Filter) and a factor graph to address these issues. We perform IESKF filtering on LiDAR and inertial measurement unit (IMU) data at the front-end to achieve a more accurate estimation of local pose and incorporate the resulting laser inertial odometry into the back-end factor graph. Furthermore, we introduce a GPS signal filtering method based on GPS state and confidence to ensure that abnormal GPS data is not used in the back-end processing. In the back-end factor graph, we incorporate loop closure factors, IMU preintegration factors, and processed GPS factors. We conducted comparative experiments using the publicly available KITTI dataset and our own experimental platform to compare the proposed SLAM system with two commonly used SLAM systems: the filter-based SLAM system (FAST-LIO) and the graph optimization-based SLAM system (LIO-SAM). The experimental results demonstrate that the proposed SLAM system outperforms the other systems in terms of localization accuracy, especially in cases of GPS signal interruption.

Keywords: autonomous mobile robots; Global Positioning System (GPS); LiDAR; inertial measurement unit (IMU); Iterative Error State Kalman Filter (IESKF); Simultaneous Localization and Mapping (SLAM)

check for updates

Citation: You, B.; Zhong, G.; Chen, C.; Li, J.; Ma, E. A Simultaneous Localization and Mapping System Using the Iterative Error State Kalman Filter Judgment Algorithm for Global Navigation Satellite System. *Sensors* **2023**, *23*, 6000. https://doi.org/10.3390/s23136000

Academic Editor: Sameh Nassar

Received: 14 May 2023
Revised: 15 June 2023
Accepted: 26 June 2023
Published: 28 June 2023

1. Introduction

With the continuous development of outdoor mobile robots, research focuses mainly on two directions: autonomous mobile robots and teleoperated mobile robots [1]. In the field of autonomous mobile robots, relying on a single sensor is no longer sufficient to meet the accuracy and robustness requirements of localization and mapping systems. The integration of multiple sensors and the fusion of their data are core technologies for achieving high-precision positioning and navigation in autonomous robots [2]. Among the sensors commonly used in outdoor scenarios, the GPS global navigation satellite system (GPS GNSS) and GNSS/INS (Inertial Navigation System) are widely utilized. Generally, GPS can provide centimeter-level positioning. However, due to the presence of urban high-rise buildings obstructing the surrounding area, GPS signals often result in inaccurate positioning [3]. Consequently, the upper computer may provide code differences or single-point positioning instead of fixed solutions. Therefore, it is common practice to either resort to teleoperation for remote control [4] or incorporate various types of sensors such as IMU, LiDAR, and cameras to assist with positioning.

Recently, numerous solutions have been proposed to address the localization problem in autonomous mobile robots and unmanned vehicles. One of the most widely adopted approaches is the Extended Kalman Filter [5], which combines GPS and IMU data fusion to estimate the localization position. However, this solution heavily relies on GPS data, and the unavailability of GPS signals can introduce significant errors in the localization results.

To overcome this challenge, computer vision and LiDAR based Simultaneous Localization and Mapping techniques are extensively utilized for state estimation in mobile robots. Compared to cameras, LiDAR can directly capture 3D structural information within a specific radius and is less sensitive to lighting conditions. Therefore, laser-based SLAM enhances system robustness and localization accuracy compared to vision-based SLAM. However, due to its dependence on the surrounding environment, LiDAR can experience drift during fast motion, leading to loss of robot positioning. Researchers addressing this issue commonly employ filtering and graph optimization methods to fuse GPS, LiDAR, and IMU data.

Gao [6] employed an Extended Kalman Filter (EKF) to estimate the positional attitude by loosely coupling GPS and LiDAR, periodically correcting the IMU. Shamsudin [7] utilized a Rao-Blackwellized Particle Filter (RBPF) to integrate GPS and LiDAR data for detecting consistency in petrochemical enterprise maps and constructing maps using GPS and LiDAR data in Fast-SLAM. Abdelaziz [8] implemented SLAM based on a loosely coupled EKF, INS, and LiDAR, matching relative poses of 3D probability maps. The proposed method was tested on the KITTI dataset to validate its robustness. Aboutaleb [9] utilized the EKF with LiDAR data, GNSS, and simplified 3D inertial sensors, employing LO to constrain system positioning drift. LIOM [10] utilized a CNN segmentation network to remove dynamic objects, fusing LiDAR and IMU data using ESKF, and generating a static global map through scan matching. LINS [11] was the first method to employ the Iterated Error State Kalman Filter (IESKF) for tightly-coupled LiDAR and IMU-based robot motion estimation. It recursively corrected the robot's estimated pose using LiDAR-extracted features, preventing filter divergence during long-term operation while maintaining computational speed. FAST-LIO [12] fused LiDAR point cloud features with IMU data using IESKF, and proposed a dimension-dependent Kalman gain formulation to reduce computational intensity resulting from laser point cloud features. FAST-LIO2 [13] also utilized IESKF and improved positioning by incorporating the original LiDAR point cloud information into the map, enhancing accuracy by observing subtle environmental features. The system employed an incremental K-D tree (IKD tree) data structure for incremental updates and dynamic point cloud smoothing, reducing computational requirements and increasing real-time performance. Li [14] proposed a bilateral haptic teleoperation method to enhance robot adaptability. Faster-LIO [15] replaced the IKD tree with an incremental voxel-based point cloud data structure (iVox) that supports incremental insertion and parallel approximation queries, resulting in an efficient and robust LiDAR inertial odometry (LIO) framework.

Another approach to address the localization problem is to utilize graph optimization [16] in the framework of SLAM algorithms. Although the aforementioned filter-based approach offers significant advantages in terms of localization speed, it falls short of the localization accuracy achieved by graph optimization.

Kukko [17] collected 3D point cloud data from the surrounding forest environment using mobile laser scanning and combined the results with GNSS/INS to optimize the trajectory using a graph optimization method. This approach accurately extracted the forest map and tree parameters. Hess [18] implemented 2D LiDAR localization and mapping based on graph optimization, employing the branch-and-bound method for scan calculation. Chang [19] proposed IMU and ODO pre-integration that incorporates odometry. They used a 3D probabilistic map at the front end to enhance point cloud matching effectiveness in feature-limited environments and ensure system trajectory accuracy when GNSS signals are unavailable. Pierzchała [20] employed a graph-based SLAM system with a 16-line laser rangefinder, camera, IMU, and GPS to evaluate relative distances between wood structures and trees. Google Cartographer [21] achieved the fusion of GNSS, 3D-LiDAR, and IMU data based on graph optimization. This method assumed that the mobile robot moved at a low, uniform speed and used gravity to solve horizontal attitude, neglecting modeling of IMU deviation errors. LOAM [22], proposed by Zhang and Singh, extracted edge and planar features from LiDAR point clouds to reduce computational effort required for matching. The framework used a high-frequency LiDAR odometry for positional estimation at the front end and low-frequency map optimization at the back end for map-building. This resulted in a low-computation, low-drift, and real-time SLAM system. LOAM consistently performed well in the KITTI [23] dataset. Subsequently, several variants and updated versions of LOAM were proposed, including LeGO-LOAM [24], R-LOAM [25], and F-LOAM [26]. These SLAM algorithms primarily focused on improving the processing time of the LOAM algorithm. For in-vehicle navigation systems, the integration of LiDAR SLAM with GNSS/INS plays a crucial role in achieving system redundancy and robustness. Shan's proposed LIO-SAM [27] utilized point cloud feature extraction with key frames at the front end to reduce computational effort. It also incorporated IMU pre-integration factor, GPS factor, and laser odometry factor into the back-end factor map optimization to construct P^3-LOAM [28]. The system combines LiDAR-SLAM with GNSS precise point positioning and estimates the covariance of laser SLAM based on the error propagation model of the SVD Jacobi matrix. Additionally, it relies on laser SLAM when the GNSS observations contain significant errors, eliminating PPP outliers and achieving high-accuracy positioning in urban canyon environments.

In summary, filter-based methods primarily rely on the first-order Markov assumption, where the current state depends only on the previous state. While this reduces computational complexity, it also leads to accumulated drift errors. On the other hand, graph optimization-based methods store the states of all previous time steps using keyframes and continuously correct accumulated errors with subsequent observations. However, the drawback is that it requires more computational resources and memory usage.

To address these challenges, this paper proposes a SLAM system that combines filtering and graph optimization, aiming to improve localization speed while maintaining accuracy. The main contributions of this work are as follows:

1. We employ the IESKF algorithm to achieve tight coupling of laser rangefinder and IMU data, enabling robust pose estimation. This approach effectively integrates sensor data to improve the accuracy and stability of frontend pose estimation.
2. We introduce a method for filtering out anomalous GPS data based on GPS state variables and confidence. This method effectively reduces the interference and errors caused by GPS data, thereby enhancing the localization performance of the SLAM system under GPS interruption.
3. We utilize factor graph optimization to fuse the frontend-generated odometry, IMU, GPS, and loop closure detection modules. By constructing a factor graph, the system can leverage the information from each module during the optimization process, improving the overall localization accuracy and consistency of the SLAM system.
4. The proposed SLAM framework's localization accuracy improvements are tested and evaluated using GPS uninterrupted/interrupted tests on the KITTI dataset and our own experimental platform.

2. SLAM System

The framework for the GPS, IMU, and LiDAR localization and mapping system presented in this paper is illustrated in Figure 1. The system comprises several modules, including data preprocessing, front-end IESKF odometry, back-end factor graph global pose optimization, GPS data filtering, and loop closure detection. The front-end utilizes a tightly coupled laser inertial odometry based on IESKF, while the back-end employs a factor graph (shown in Figure 2) to fuse the front-end LIO factors, IMU predicted measurement factors, and GPS factors.

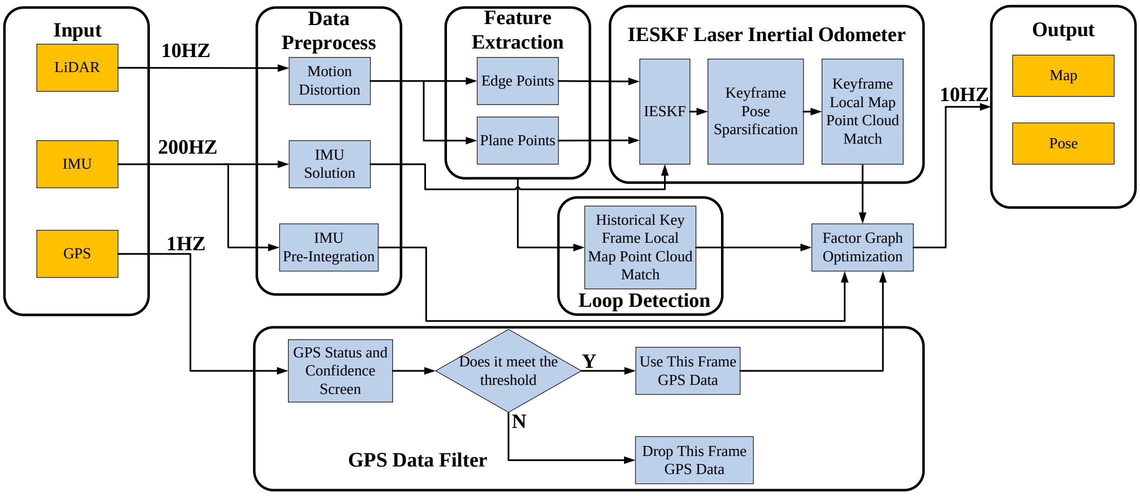


Figure 1. SLAM system framework diagram.

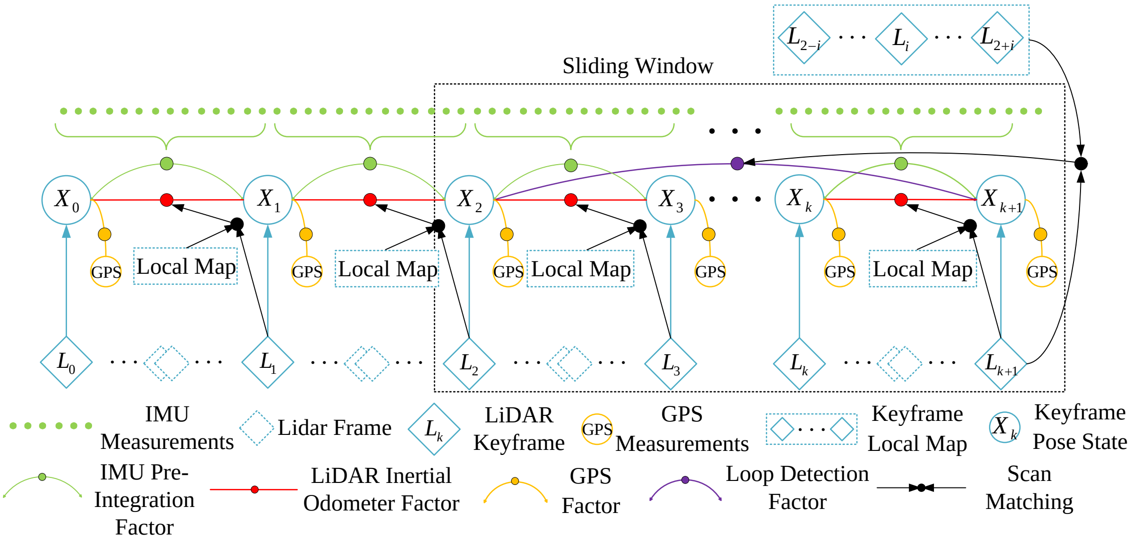


Figure 2. Factor graph model graph.

Factor Graph Optimization is a graph model used for probabilistic inference and parameter estimation problems. It employs a graph structure composed of nodes and edges to represent the dependencies and constraints between variables. In this model, nodes represent the variables to be optimized, while edges represent the constraints. By constructing such a graph, we can gain a better understanding of the problem's structure and utilize optimization algorithms to find the optimal variable configuration. Factor Graph Optimization provides a flexible and effective approach for addressing complex inference and estimation problems.

The process of Factor Graph Optimization involves iteratively minimizing an objective function that measures the difference between the predicted state vector and the actual state vector. In this paper, we employ Incremental Smoothing and Mapping (iSAM2) [29] using Bayesian tree mapping for the factor graph optimization process. The objective of this process is to obtain the posterior pose distribution of the robot, given the known sensor measurement noise, and further model it by iteratively decomposing a set of factors $\phi(\mathbf{X})$. The objective function is described by Equation (1):

$$\arg\min_{\mathbf{X}} f(\mathbf{X}) = \arg\max_{\mathbf{X}} \prod_i \phi_i(\mathbf{X}_i) \tag{1}$$

where $\mathbf{X}$ is the vector to be estimated, and $f(\mathbf{X})$ is the cost function.

Equation (1) can be equivalently expressed as a least squares form, as shown in Equation (2):

$$\arg\min_{\mathbf{X}} f(\mathbf{X}) \triangleq \arg\min_{\mathbf{X}} \sum_{i} \|h_i(\mathbf{X}_i) - \mathbf{z}_i\|^2_{\mathbf{D}_i} \tag{2}$$

where $h(\mathbf{X})$ is the observation equation, $\mathbf{z}$ is the observed value, and $\mathbf{D}$ is the covariance matrix of the observed value.

The modeling of the robot state vector $\mathbf{x}$ consists of the rotation matrix $\mathbf{R} \in SO(3)$, position $\mathbf{p} \in \mathbb{R}^3$, velocity $\mathbf{v}$, and IMU bias $\mathbf{b}$. A transformation $\mathbf{T} \in SE(3)$ from the robot base O to the world frame W is represented as $\mathbf{T}$=$[\mathbf{R} \mid \mathbf{p}]$.

$$\mathbf{x} = \left[\mathbf{R}^{\mathbf{T}}, \mathbf{p}^{\mathbf{T}}, \mathbf{v}^{\mathbf{T}}, \mathbf{b}^{\mathbf{T}}\right]^{\mathbf{T}} \tag{3}$$

2.1. IESKF LiDAR Inertial Odometry Factor

In this paper, we employ an IESKF in the front-end to achieve tight coupling of sensor data from laser odometry and IMU, aiming to achieve higher algorithm accuracy than LIO-SAM when GPS is interrupted. Compared to traditional Kalman filter methods, this approach offers three main advantages [30]. Firstly, it reduces computational complexity by ignoring second-order products, resulting in a smaller error state. Secondly, it addresses parameterization and gimbal lock problems by keeping the orientation error state small. Thirdly, the slow change in the error state allows for error correction at a lower rate than prediction. The IESKF filtering method used in this paper is based on the approaches presented in LINS and FAST-LIO, and the algorithm steps are outlined below:

(1) Input the posterior state variables $\overset{\frown}{x}_k$ and covariance matrix $\overset{\frown}{P}_k$ output by the previous IESKF, the laser point cloud after motion compensation, and the IMU data collected during the current laser scan.

(2) Predict the state variables and covariance matrix as shown in Equations (4) and (5). In Equation (4): $\oplus$ denotes the generalized addition; $\breve{x}_{i+1}$ and $\breve{x}_k$ respectively represent the prior system state variables between the laser frames k-th and $k+1$-th when receiving IMU data at times i and $i+1$; $T = t_{i+1} - t_i$ denotes the IMU sampling period; $f(\breve{x}_i, u_i, \omega_i)$ denotes the system state transition matrix; and u_i and ω_i represent the IMU measurement values and their measurement noise at time i. In Equation (5): $\breve{P}_{i+1}$ represents the predicted covariance matrix at time $i+1$; F_i represents the predicted state matrix at time i; B_i represents the noise matrix; Q represents the noise covariance matrix; and $\overset{\frown}{P}_k$ represents the posterior covariance matrix of the laser k-th frame.

$$\begin{aligned} \breve{x}_{i+1} &= \breve{x}_i \oplus [T \cdot f(\breve{x}_i, u_i, \omega_i)] \\ \breve{x}_0 &= \overset{\frown}{x}_k \end{aligned} \tag{4}$$

$$\begin{aligned} \breve{P}_{i+1} &= F_i \breve{P}_i F_i^{\mathrm{T}} + B_i Q B_i^{\mathrm{T}} \\ \breve{P}_0 &= \overset{\frown}{P}_k \end{aligned} \tag{5}$$

(3) Setting the initial value of the iteration count α to 1, the state quantity of the iteration is $\breve{x}^{\alpha=0}_{k+1} = \breve{x}_{k+1}$.

(4) Judge whether the absolute value of the difference between the state quantity obtained after one iteration and the previous iteration is less than the threshold ∂, represented by the symbol in Formula (6), where $\ominus$ denotes generalized subtraction. If it is less than the threshold ∂, then repeat the following loop.

$$\left\| \breve{x}^{\alpha+1}_{k+1} \ominus \breve{x}^{\alpha}_{k+1} \right\| < \partial \tag{6}$$

(a) Calculate the Jacobian matrix J^{α}_{k+1} of the error state vector at $\delta x^{\alpha}_{k+1} = 0$ point using Formula (7), where δx_{k+1} represents the error state vector of $k+1$-th frame. Use Formula (8) to update the prior covariance matrix $\overset{\smile}{P}_{k+1}$ during the iteration process.

$$\begin{aligned} \delta x_{k+1} &= \overset{\smile}{x}^{\alpha}{}_{k+1} \ominus \overset{\smile}{x}_{k+1} + J^{\alpha}_{k+1}\delta x^{\alpha}_{k+1} \\ J^{\alpha}_{k+1} &= \begin{bmatrix} A_{k+1}(\delta\theta_{k+1})^{-\mathrm{T}} & 0_{3\times 15} \\ 0_{15\times 3} & I_{15\times 15} \end{bmatrix} \end{aligned} \tag{7}$$

$$\overset{\smile}{P}_{k+1} = \left(J^{\alpha}_{k+1}\right)^{-1}\overset{\smile}{P}_{k+1}\left(J^{\alpha}_{k+1}\right)^{-\mathrm{T}} \tag{8}$$

(b) Transform the laser point cloud into the world coordinate system, and calculate the residual equation $f(x^{k}_{k+1})$ and covariance matrix H_{k+1} of the observation using Formulas (9) and (10), respectively. Here, $\widetilde{X}^{L_e}{}_{(k+1,i)}$ and $\widetilde{X}^{L_s}{}_{(k+1,i)}$ represent the coordinate sets of feature points after motion compensation for corner points L_e and plane points L_s, respectively, between k-th frame and $k+1$-th frame. The covariance matrix H_{k+1} is represented using the formula from LINS, R^{k}_{k+1} represents the pose transformation of the laser between k-th frame and $k+1$-th frame, and $[\bullet]_{\times}$ denotes the skew-symmetric matrix of the variable.

$$f(x^{k}_{k+1}) = \left\{ \begin{array}{l} \dfrac{\left|\left(\widetilde{X}^{L_e}{}_{(k+1,i)} - X^{L_e}_{(k+1,i)}\right)\times\left(\widetilde{X}^{L_e}{}_{(k+1,i)} - X^{L_e}_{(k,l)}\right)\right|}{\left|X^{L_e}_{(k,j)} - X^{L_e}_{(k,l)}\right|} \\ \dfrac{\left|(\widetilde{X}^{L_s}{}_{(k+1,i)} - X^{L_s}_{(k,j)})\left((X^{L_s}_{(k,j)} - X^{L_s}_{(k,l)})\times(X^{L_s}_{(k,j)} - X^{L_s}_{(k,m)})\right)\right|}{\left|\left(X^{L_s}_{(k,j)} - X^{L_s}_{(k,l)}\right)\times\left(X^{L_s}_{(k,j)} - X^{L_s}_{(k,m)}\right)\right|} \end{array} \right. \tag{9}$$

$$\begin{aligned} H_{k+1} &= \frac{\partial f}{\partial \widetilde{X}^{L}_{(k+1,i)}}\cdot\frac{\partial \widetilde{X}^{L}_{(k+1,i)}}{\partial \delta x} \\ &= \left\{ \begin{array}{l} \left\{ \dfrac{\left\{\left[\left(\widetilde{X}^{L_e}_{(k+1,i)} - X^{L_e}_{(k+1,i)}\right)\times\left(\widetilde{X}^{L_e}_{(k+1,i)} - X^{L_s}_{(k,l)}\right)\right]^{\mathrm{T}}}{\left|\left(\widetilde{X}^{L_e}_{(k+1,i)} - X^{L_e}_{(k+1,i)}\right)\times\left(\widetilde{X}^{L_e}_{(k+1,i)} - X^{L_s}_{(k,l)}\right)\right|} \right\}\cdot\left[R^{k}_{k+1}\left(X^{L_s}_{(k+1,i)}\right)_{\times}, I\right] \\ \left\{ \dfrac{\left[\left(X^{L_s}_{(k,j)} - X^{L_s}_{(k,l)}\right)\times\left(X^{L_s}_{(k,j)} - X^{L_s}_{(k,m)}\right)\right]^{\mathrm{T}}}{\left|\left(X^{L_s}_{(k,j)} - X^{L_s}_{(k,l)}\right)\times\left(X^{L_s}_{(k,j)} - X^{L_s}_{(k,m)}\right)\right|} \right\}\cdot\left[R^{k}_{k+1}\left(X^{L_s}_{(k+1,i)}\right)_{\times}, I\right] \end{array} \right. \end{aligned} \tag{10}$$

(c) Update the state variables $\overset{\smile}{x}^{\alpha+1}_{k+1}$ and Kalman gain K_{k+1} using Formulas (11) and (12).

$$\overset{\smile}{x}^{\alpha+1}_{k+1} = \overset{\smile}{x}^{\alpha}_{k+1} - K_{k+1}f^{\alpha}_{k+1} - (I - K_{k+1}H_{k+1})\left(J^{\alpha}_{k+1}\right)^{-1}\left(\overset{\smile}{x}^{\alpha}_{k+1} \ominus \overset{\smile}{x}_{k+1}\right) \tag{11}$$

$$K_{k+1} = \left(\overset{\smile}{P}^{-1}{}_{k+1} + H^{\mathrm{T}}_{k+1}L^{-1}_{k+1}H_{k+1}\right)^{-1} H^{\mathrm{T}}_{k+1}L^{-1}_{k+1} \tag{12}$$

(5) Output the posterior state quantity $\overset{\frown}{x}_{k+1}$ and posterior covariance $\overset{\frown}{P}_{k+1}$ using Equations (13) and (14).

$$\overset{\frown}{x}_{k+1} = \overset{\smile}{x}^{\alpha+1}{}_{k} \tag{13}$$

$$\overset{\frown}{P}_{k+1} = (I - K_{k+1}H_{k+1})\overset{\smile}{P}_{k+1} \tag{14}$$

During factor graph optimization, there is redundant information between IMU data and laser point cloud data. Including IESKF odometry data of each frame in the back-end factor graph optimization only leads to a slight improvement in localization accuracy but consumes a significant amount of computational resources, which ultimately affects the localization accuracy of the system. Therefore, we employ the keyframe and sliding window strategy in factor graph optimization to reduce the computational resources required in the back-end. Keyframes are selected based on representative laser-IMU odometry data over a specific period, helping to reduce the data volume. In the sliding window approach, only the keyframe data within the window is optimized, while the regular data frames are discarded.

In this paper, "ordinary frames" refers to laser frames observed by IESKF, while "keyframes" are determined based on position or attitude changes estimated by IESKF's laser inertial odometry exceeding 1 m or 5°, respectively. Adjacent keyframes are utilized to construct local maps. Based on the current keyframe pose, i nearest keyframes are extracted to form the adjacent keyframe set $\{F_{i-k}, \cdots, F_k\}$, and the poses corresponding to the adjacent keyframe set are transformed to the current keyframe F coordinate system. After the transformation, the adjacent keyframe point clouds are merged into one local map. As subsequent new keyframe point clouds are added to the local map, keyframe point clouds that are far away from the local map are removed. To obtain a more accurate pose transformation relationship between two keyframes, this paper adopts the ICP registration algorithm to match the current keyframe with the local map and derive the pose transformation relationship. The residual equation between the k-th and k+1-th laser radar keyframes can be obtained as shown in Equation (15):

$$r_L = \begin{bmatrix} \Delta t - R_k^{\mathrm{T}}(t_{k+1} - t_k) \\ \log\left(\Delta R^{\mathrm{T}} R_k^{\mathrm{T}} R_{k+1}\right) \end{bmatrix} \tag{15}$$

2.2. GPS Factor

Due to significant fluctuations in GPS data in highly urbanized environments, this paper incorporates a GPS state and confidence filtering approach to screen the available GPS data and include them as GPS factors in the factor graph. This method aims to exclude anomalous GPS data from being included in the factor graph, thereby ensuring higher accuracy. Specifically, this paper only utilizes GPS data with fixed solutions and narrow lane fixed solutions.

In geometric positioning methods, the accuracy of localization is influenced by the relative distances between multiple base stations and mobile stations, which is commonly referred to as Dilution of Precision (DOP). To calculate the DOP factor, we introduce the GPS single-point positioning model, which is represented by the following equation:

$$\rho_{\mathrm{j}} = \sqrt{(\mathrm{x_j} - \mathrm{x_u})^2 + \left(\mathrm{y_j} - \mathrm{y_u}\right)^2 + (\mathrm{z_j} - \mathrm{z_u})^2} + \mathrm{c}(\mathrm{t_u} - \mathrm{t_j}) \tag{16}$$

where $(\mathrm{x_u}, \mathrm{y_u}, \mathrm{z_u})$ is the receiver coordinates, $\left(\mathrm{x_j}, \mathrm{y_j}, \mathrm{z_j}\right)$ is the base station coordinates, $\mathrm{t_u}$ and $\mathrm{t_j}$ are the clock bias between the receiver and the base station, ρ_{j} represents the pseudorange from the receiver to the base station, and j represents the number of visible base stations.

Given the approximate values of receiver coordinates $(\widetilde{\mathrm{x_u}}, \widetilde{\mathrm{y_u}}, \widetilde{\mathrm{z_u}})$, and clock error $\widetilde{\mathrm{t_u}}$, we can linearize the positioning model by performing a first-order Taylor series expansion, as shown in Equation (17). Furthermore, we can represent Equation (17) in matrix form as Equation (18):

$$\Delta\rho_{\mathrm{j}} = \mathrm{l_j}\Delta\mathrm{x_u} + \mathrm{m_j}\Delta\mathrm{y_u} + \mathrm{n_j}\Delta\mathrm{z_u} - \mathrm{c}\Delta\mathrm{t_u} \tag{17}$$

$$\mathbf{\Delta P} = \mathbf{H}\mathbf{\Delta X} \tag{18}$$

$$\mathbf{\Delta P} = \begin{bmatrix} \Delta\rho_1 \\ \Delta\rho_2 \\ \vdots \\ \Delta\rho_{\mathrm{j}} \end{bmatrix}, \mathbf{H} = \begin{bmatrix} \mathrm{l_1} & \mathrm{m_1} & \mathrm{n_1} & 1 \\ \mathrm{l_2} & \mathrm{m_2} & \mathrm{n_2} & 1 \\ \vdots & \vdots & \vdots & \vdots \\ \mathrm{l_j} & \mathrm{m_j} & \mathrm{n_j} & 1 \end{bmatrix}, \mathbf{\Delta X} = \begin{bmatrix} \Delta\mathrm{x_u} \\ \Delta\mathrm{y_u} \\ \Delta\mathrm{z_u} \\ -\mathrm{c}\Delta\mathrm{t_u} \end{bmatrix} \tag{19}$$

where $\mathrm{l_j}$, $\mathrm{m_j}$, and $\mathrm{n_j}$ represent the direction cosines of the unit vector pointing from the approximate position towards the j-th base station.

By applying the least squares method to solve Equation (18), we can derive quantitative expressions for the components of the symmetric matrix **G**, which represent the accuracy factors. These expressions are given by the following equation:

$$\mathbf{G} = \left(\mathbf{H}^{\mathbf{T}}\mathbf{H}\right)^{-1} = \begin{bmatrix} \mathbf{g}_{11} & \mathbf{g}_{12} & \mathbf{g}_{13} & \mathbf{g}_{14} \\ \mathbf{g}_{12} & \mathbf{g}_{22} & \mathbf{g}_{23} & \mathbf{g}_{24} \\ \mathbf{g}_{13} & \mathbf{g}_{23} & \mathbf{g}_{33} & \mathbf{g}_{34} \\ \mathbf{g}_{14} & \mathbf{g}_{24} & \mathbf{g}_{34} & \mathbf{g}_{44} \end{bmatrix} \tag{20}$$

We select the square root of the sum of squared errors in dimensions, precision, and elevation as the confidence criterion, which is commonly known as PDOP (Position Dilution of Precision). It can be calculated using the following equation:

$$\mathbf{PDOP}\text{=}\sqrt{\mathbf{g}_{11}\text{+}\mathbf{g}_{22}\text{+}\mathbf{g}_{33}} \tag{21}$$

First, the inspection robot is moved to an open area to allow for movement, and multiple confidence values are recorded when the GPS state is a fixed solution. The maximum value among these recorded confidences is selected as the threshold for fixed solution confidence. Second, the steps mentioned above are repeated to obtain the confidence threshold for narrow lane fixed solutions. Finally, during the operation of the SLAM system, the corresponding confidence threshold is selected based on the GPS state. This threshold is then compared with the current confidence value, and GPS data with a confidence lower than the threshold are considered usable. Figure 3 illustrates the flowchart of the GPS state and confidence filtering strategy employed in this paper.

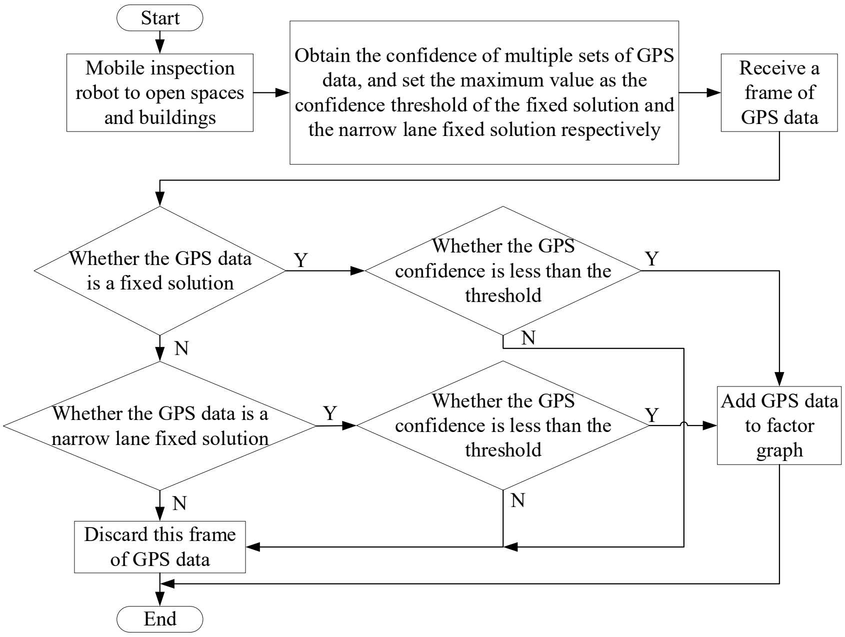

Start
Mobile inspection robot to open spaces and buildings
Obtain the confidence of multiple sets of GPS data, and set the maximum value as the confidence threshold of the fixed solution and the narrow lane fixed solution respectively
Receive a frame of GPS data
Whether the GPS data is a fixed solution
Y
Whether the GPS confidence is less than the threshold
Y
N
N
Whether the GPS data is a narrow lane fixed solution
Y
Whether the GPS confidence is less than the threshold
Y
Add GPS data to factor graph
N
N
Discard this frame of GPS data
End

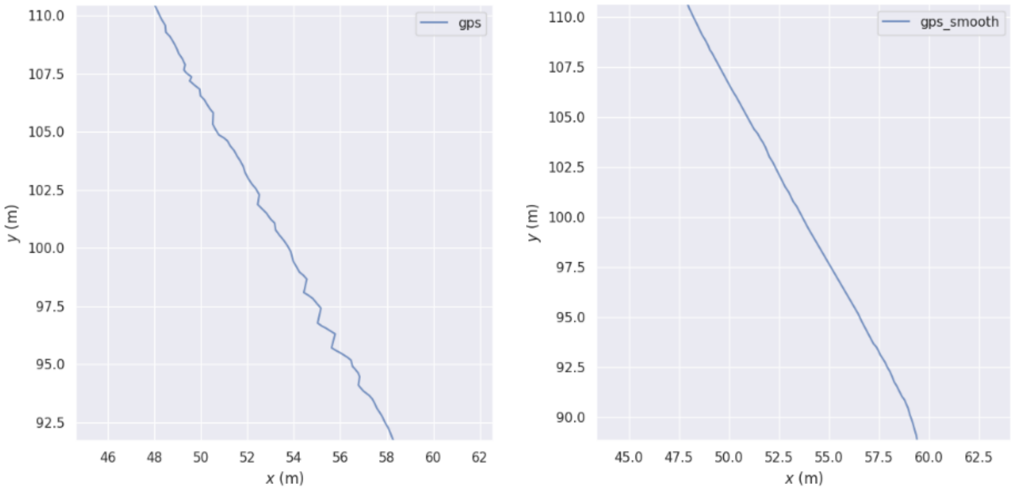
Figure 3. Flow chart of screening strategies based on GPS status and confidence.

After filtering out unreliable data based on confidence, we utilize the remaining GPS data to calculate the coordinates in the W-frame for latitude, longitude, and altitude. These coordinates are then incorporated into the factor graph as a GPS position constraint cost function. The GPS position constraint cost function is represented by the following equation:

$$\sum_{\mathrm{i}\in\mathbb{G}} \left\| \mathrm{e}\left(R_{t_i}^{o}, R_{o}^{w}\right) \right\|_{D_g}^{2} = \sum_{\mathrm{i}\in\mathbb{G}} \left\| R_{o}^{w}\left(R_{t_i}^{o} T_{g}^{b}\right) - T_{g}^{w} \right\|_{D_g}^{2} \tag{22}$$

where $R_{t_i}^{o}$ is the pose of the IMU in the vehicle coordinate system at time t_i, R_{o}^{w} is the transformation parameters between the vehicle coordinate system and the global coordinate system W, T_g^w is the GNSS positioning result in the global coordinate system W, T_g^b is the antenna lever arm for GNSS, D_g is the variance-covariance matrix for T_g^w provided by the GNSS RTK positioning solution, and $\mathbb{G}$ is the set of nodes with GNSS position correction.

We conducted tests in practical scenarios, specifically at the rear of the supporting service center building in the Intelligent Technology Park, where GPS data exhibited fluctuations. Figure 4 illustrates the GPS trajectories before and after applying our filtering approach. It can be observed that the GPS data exhibits significant fluctuations prior to filtering, whereas the fluctuations are reduced after the application of our filtering method.



(**a**) Before GPS data filtering (**b**) After GPS data filtering

Figure 4. Filter GPS comparison chart based on GPS status and confidence.

2.3. Loop Detection Factor

Similar to LIO-SAM, this paper employs a keyframe-based Euclidean distance approach for loop closure detection. Firstly, the laser point cloud is transformed into the world coordinate system. Based on the position of the current keyframe, a search range of distance d is defined to identify historical keyframes that are in close proximity and have a longer detection time. The positions of the keyframes within the search range are further filtered based on a specified time interval. The local feature point cloud map is constructed by aggregating feature point clouds from a range of 25 frames centered around the identified historical keyframes. The current keyframe is then matched with the local feature point cloud map using ICP point cloud registration to determine the relative pose transformation relationship in the world coordinate system. For a more comprehensive explanation of the loop closure detection process, please refer to LIO-SAM.

2.4. IMU Pre-Integration

The angular velocity and acceleration measured from the IMU are defined as follows:

$$\widehat{\omega}_t = \omega_t + b_t^{\omega} + n_t^{\omega} \tag{23}$$

$$\widehat{a}_t = R_t^{BW}(a_t - g) + b_t^a + n_t^a \tag{24}$$

where $\widehat{\omega}_t$ and $\widehat{a}_t$ are the measurements of the IMU at the moment t and the B coordinate systems, $\widehat{\omega}_t$ and $\widehat{a}_t$ are subject to the slowly transformed bias b_t and white noise n_t, R_t^{BW} is the rotation matrix from the coordinate system W to B, and g is a fixed gravity vector in the W coordinate system.

The robot motion was then inferred from the IMU measurements. The position, attitude, and velocity of the robot during $t + \Delta t$ were calculated as follows:

$$v_{t+\Delta t} = v_t + g\Delta t + R_t(\widehat{a}_t - b_t^a - n_t^a)\Delta t \tag{25}$$

$$p_{t+\Delta t} = p_t + v_t \Delta t + \frac{1}{2} g \Delta t^2 + \frac{1}{2} R_t (\widehat{a}_t - b_t^a - n_t^a) \Delta t^2 \tag{26}$$

$$R_{t+\Delta t} = R_t \exp((\widehat{\omega}_t - b_t^\omega - n_t^\omega) \Delta t) \tag{27}$$

where $R = R_t^{WB} = R_t^{BWT} R_t^{BW}$ are the rotation matrices of the coordinate systems B to W, and the angular velocity and acceleration of the base coordinates B are assumed to be constant during the integration process. Then, we use the IMU pre-integration method to obtain the relative motion of the carriers within adjacent timestamps, where the pre-integrated measurements Δv_{ij}, Δp_{ij}, and ΔR_{ij} between moments i and j are calculated by the following equations:

$$\Delta v_{ij} = {R_i}^T (v_j - v_i - g \Delta t_{ij}) \tag{28}$$

$$\Delta p_{ij} = {R_i}^T (p_j - p_i - v_i \Delta t_{ij} - \frac{1}{2} g {\Delta t_{ij}}^2) \tag{29}$$

$$\Delta R_{ij} = {R_i}^T R_j \tag{30}$$

3. Experimental

3.1. KITTI Dataset Testing and Evaluation

This paper presents an experimental study on the fusion of filtering and graph optimization SLAM algorithms using the KITTI datasets 05, 07, and 10. The trajectories and ATE statistical indicators generated are compared and analyzed with an open-source multi-sensor fusion SLAM algorithm using the EVO [31] trajectory evaluation tool. The filtering algorithm used in this experiment is FAST-LIO, while the graph optimization algorithm used is LIO-SAM. To evaluate the robustness of the algorithm when GPS is suddenly interrupted, GPS data is also interrupted. Since FAST-LIO does not incorporate GPS, its accuracy is not affected by GPS interruption. Therefore, its ATE statistical indicator is not presented in the GPS interruption experiment.

3.1.1. GPS Uninterrupted Experiment

To start, we performed the GPS non-interruption experiment and obtained trajectories for sequences 05, 07, and 10, as depicted in Figures 5–8, respectively. Our algorithm exhibited superior trajectory results compared to the two open-source algorithms, with trajectories that closely aligned with the true values.

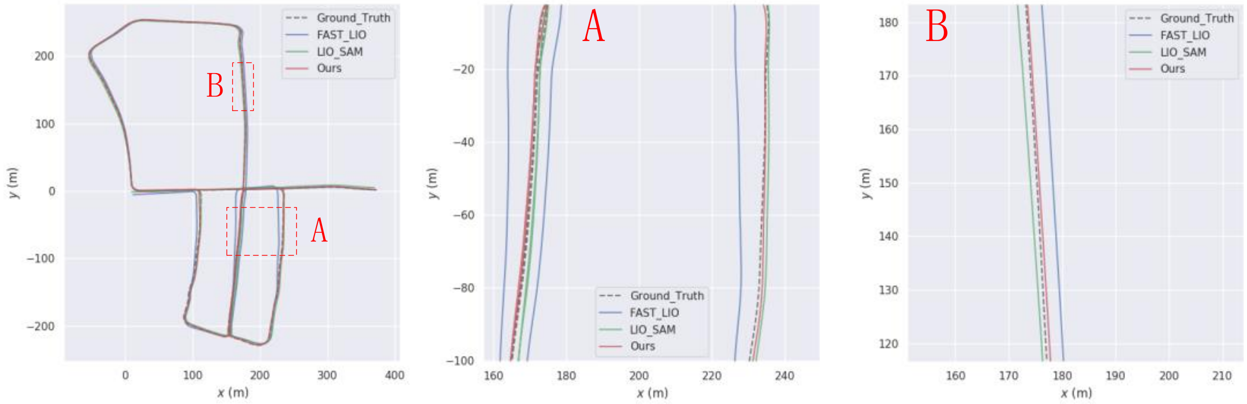


Figure 5. Comparison of our trajectory with LIO-SAM and FAST-LIO on the KITTI05 sequence. (**A**) and (**B**) are the sections where our trajectory is better than LIO-SAM and FAST-LIO.

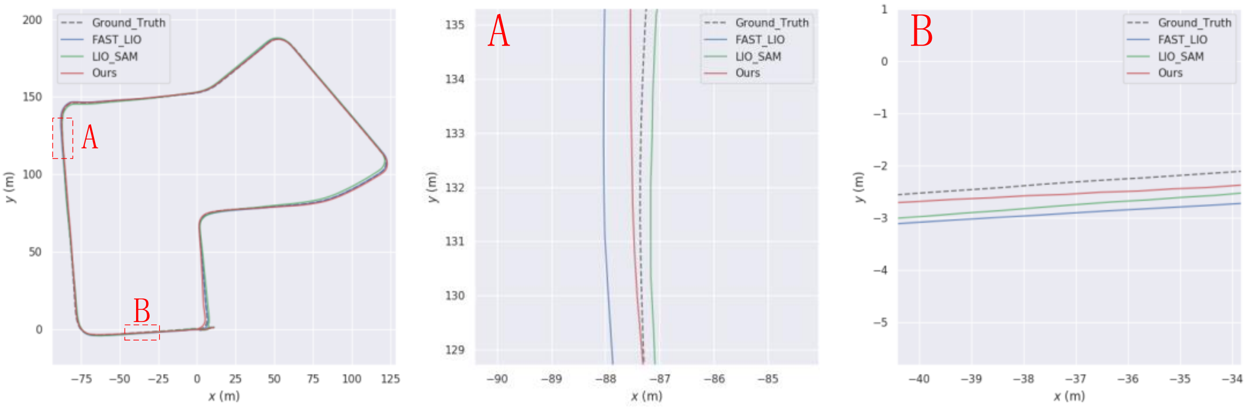


Figure 6. Comparison of our trajectory with LIO-SAM and FAST-LIO on the KITTI07 sequence. (**A**) and (**B**) are the sections where our trajectory is better than LIO-SAM and FAST-LIO.

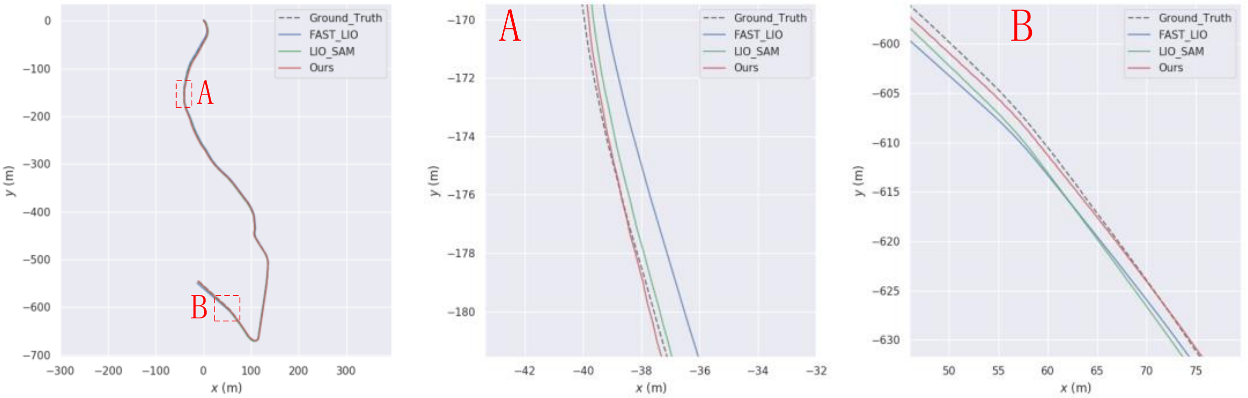


Figure 7. Comparison of our trajectory with LIO-SAM and FAST-LIO on the KITTI10 sequence. (**A**) and (**B**) are the sections where our trajectory is better than LIO-SAM and FAST-LIO.

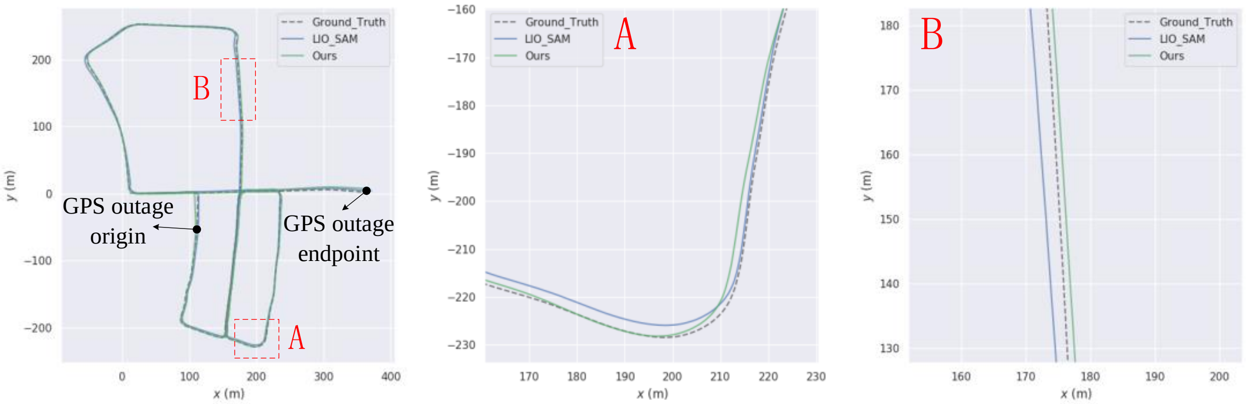


Figure 8. Comparison of our trajectory with LIO-SAM on the KITTI05 sequence. (**A**) and (**B**) are the road sections where our trajectory is better than LIO-SAM.

In Table 1, we present our algorithm alongside the ATE statistical data for FAST-LIO and LIO-SAM.

In terms of performance metrics on the 05 sequence, the algorithm proposed in this paper outperformed both LIO-SAM and FAST-LIO. Specifically, it achieved improvements of 21.57%, 44.75%, 43.19%, and 12.22% in maximum error, average error, root mean square error, and minimum error, respectively, when compared to LIO-SAM. Furthermore, compared to FAST-LIO, the proposed algorithm showed improvements of 74.09%, 79.64%, 63.68%, and 58.57% in the same metrics.

Table 1. Comparing the absolute error of our method, LIO-SAM and FAST-LIO on the KITTI dataset.

Sequence	Method	Max (m)	Mean (m)	Median (m)	Min (m)
05	Ours	1.572958	**0.555240**	**0.623827**	**0.054046**
	LIO-SAM	**1.140164**	0.604368	0.666729	0.050848
	FAST-LIO	12.143285	5.453287	6.046867	0.743559
07	Ours	**1.333466**	**0.555258**	**0.605827**	**0.079949**
	LIO-SAM	**1.323349**	0.617591	0.665901	0.147816
	FAST-LIO	4.248161	2.376953	2.522683	0.456325
10	Ours	**2.751671**	**1.146823**	**1.293698**	0.262802
	LIO-SAM	5.528663	1.544244	1.914991	**0.140108**
	FAST-LIO	5.772416	2.933960	3.189745	0.497090

On the 07 sequence, the algorithm presented in this paper did not perform as well as LIO-SAM in terms of minimum error. However, it outperformed both LIO-SAM and FAST-LIO in other statistical indicators. Specifically, it achieved improvements of 20.14% and 52.19% in maximum error, 30.95% and 61.28% in average error, and 43.47% and 61.27% in root mean square error, respectively.

Finally, on the 10 sequence, the algorithm proposed in this paper outperformed both LIO-SAM and FAST-LIO in all metrics. In particular, it achieved improvements of 32.42% and 50.24% in maximum error, 48.09% and 58% in average error, 46.28% and 57.9% in root mean square error, and 65.88% and 47.13% in minimum error, respectively.

3.1.2. GPS Interrupted Experiment

This paper aims to simulate sudden interruptions of GPS signals by applying interruption processing to GPS data in the KITTI dataset, starting from the 20th second. Specifically, interruptions of approximately 268 s, 95 s, and 107 s were applied to the KITTI05, KITTI07, and KITTI10 sequences, respectively. The trajectories of these sequences are shown in Figures 8–10, respectively. From Figures A and B in each of these figures, it can be observed that the trajectory generated by the algorithm proposed in this paper is superior to that generated by LIO-SAM in GPS signal interruption scenarios.

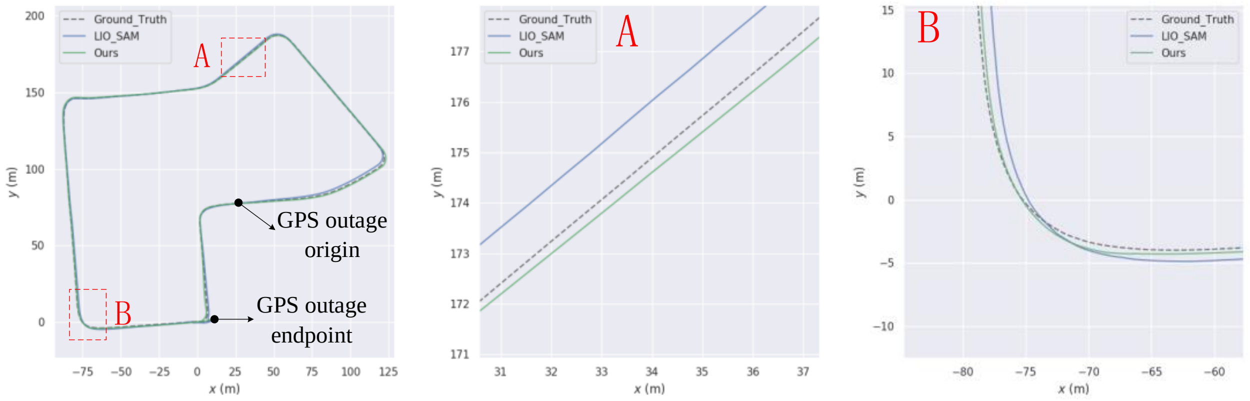


Figure 9. Comparison of our trajectory with LIO-SAM on the KITTI07 sequence. (**A**) and (**B**) are the road sections where our trajectory is better than LIO-SAM.

Table 2 presents the ATE statistical data for the proposed algorithm and LIO-SAM in GPS signal interruption scenarios.

For the KITTI05 sequence, the proposed algorithm has a slightly higher minimum error value than LIO-SAM, with an increase of 0.126808 m. However, the proposed algorithm outperforms LIO-SAM in all other statistical indicators, with improvements of 40.92%, 29.01%, and 28.2%, respectively.

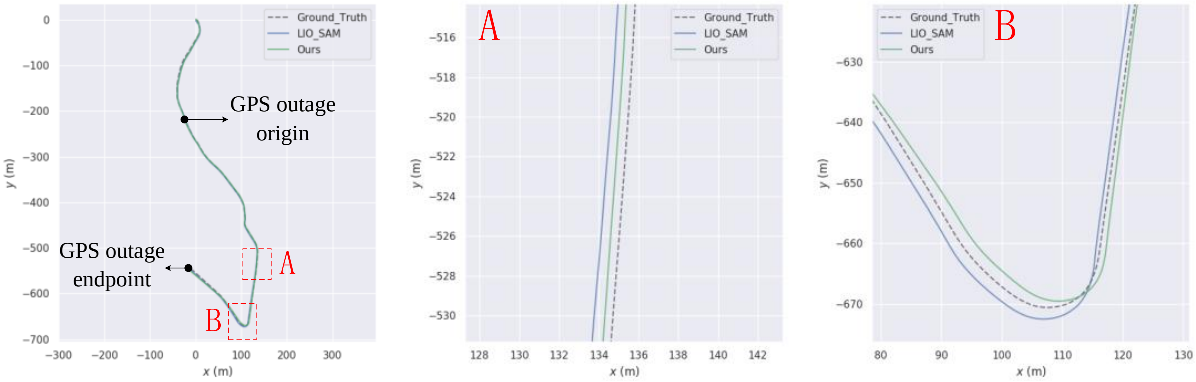


Figure 10. Comparison of our trajectory with LIO-SAM on the KITTI10 sequence. (**A**) and (**B**) are the road sections where our trajectory is better than LIO-SAM.

Table 2. Comparing the absolute error of our method with LIO-SAM on KITTI.

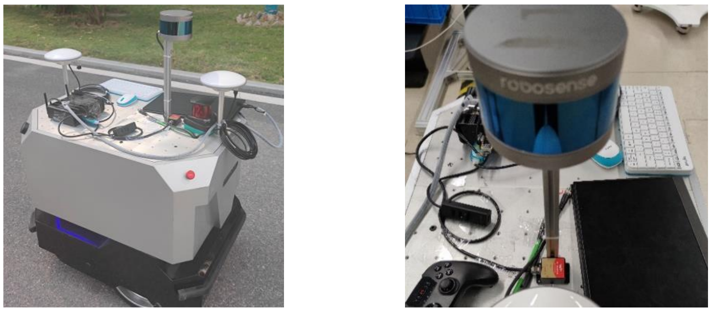
Sequence	Method	Max (m)	Mean (m)	Median (m)	Min (m)
05	Ours	**2.936086**	**1.242616**	**1.392347**	0.177911
	LIO-SAM	4.969321	1.750427	1.939206	0.051103
07	Ours	**1.272413**	**0.667964**	**0.723489**	**0.139385**
	LIO-SAM	2.237385	0.775331	0.872355	**0.135207**
10	Ours	**2.978839**	**1.422265**	**1.524391**	0.331625
	LIO-SAM	6.073832	1.803528	2.169900	**0.122156**

For the KITTI07 sequence, the proposed algorithm has a slightly higher minimum error value than LIO-SAM, but it outperforms LIO-SAM in all other statistical indicators, with improvements of 43.13%, 13.85%, and 17.06%, respectively.

For the KITTI10 sequence, the proposed algorithm has a slightly higher minimum error value than LIO-SAM, with an increase of 0.209469 m. However, the proposed algorithm outperforms LIO-SAM in all other indicators, with improvements of 50.96%, 21.14%, and 29.75%, respectively.

3.2. Experiments on a Real Platform

In a subsequent evaluation of the method proposed in this paper, we conducted localization map-building experiments on an actual autonomous mobile robot in an outdoor environment. All datasets of real scenes in the park were collected on an experimental platform, as shown in Figure 11. The experimental platform was equipped with a 1.60 GHz Intel i5-8250U IPC and connected to an RS-LiDAR-16 LiDAR with a frequency of 10 Hz, a Witt Smart IWT905 IMU with a frequency of 200 Hz, and a GPS consisting of a 10 Hz BeiDou XingTong base station NC502-D and a mobile station NC507-S.



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

Figure 11. (**a**) Outdoor autonomous mobile robot experimental platform equipped with GPS, LiDAR, and IMU sensors for data collection; and (**b**) closer view of the LiDAR and IMU sensors.

An experiment was conducted to evaluate the positioning accuracy of our algorithm applied to data collected with a mobile autonomous robot in an outdoor environment where fluctuations or interruptions occurred in the GPS data. A navigation area with an approximate size of 280 × 280 m^2 was created in the park, and the resulting map is presented in Figure 12.

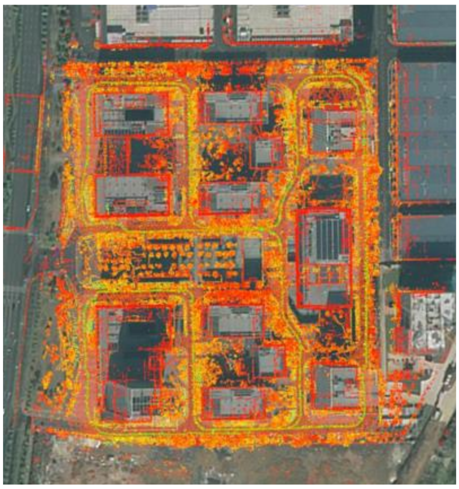


Figure 12. Point cloud map built on our own dataset using our method.

3.2.1. GPS Uninterrupted Experiment

First, an experiment was conducted with uninterrupted GPS, and the resulting trajectory within the intelligent technology park is illustrated in Figure 13. The trajectory generated by the algorithm proposed in this paper closely aligns with the true trajectory, demonstrating superior trajectory accuracy compared to LIO-SAM. For instance, in the road section depicted in Figure 13A, the algorithm proposed in this paper exhibits the closest match to the true trajectory. Similarly, in the road section shown in Figure 13B, the proposed algorithm outperforms both LIO-SAM and FAST-LIO.

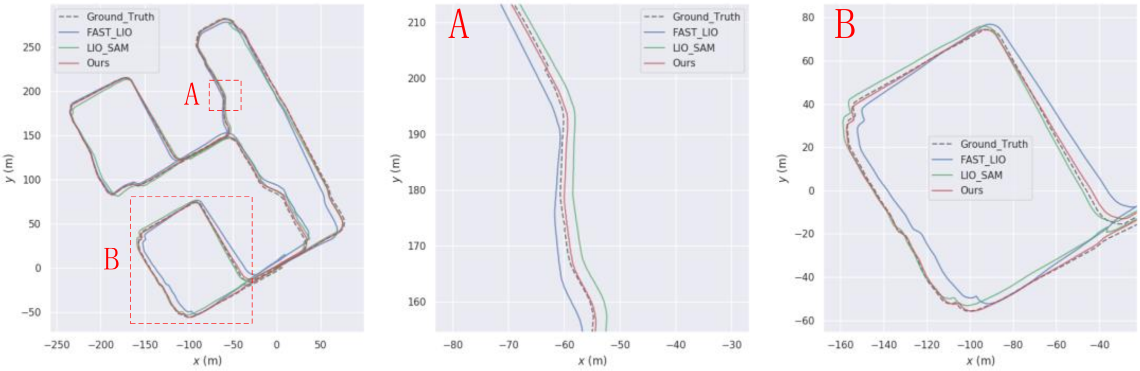


Figure 13. Comparison charts with LIO-SAM trajectory on our own data set (**A**) and (**B**) are road sections where our trajectory is better than LIO-SAM.

Within the intelligent technology park, Table 3 presents the ATE statistical data comparing the algorithm proposed in this paper, FAST-LIO, and LIO-SAM. In all statistical indicators, the algorithm proposed in this paper outperforms the two mentioned open-source algorithms. The maximum error value has been improved by 51.03% and 64.64%, the average error value has been improved by 21.68% and 55.5%, the root mean square error has been improved by 24.98% and 58.58%, and the minimum error value has been improved by 71.30% and 68.92%, respectively. These results affirm the high accuracy of the algorithm proposed in this paper.

Table 3. Comparing the absolute error of our method with LIO-SAM on our own dataset.

Method	Max (m)	Mean (m)	Median (m)	Min (m)
Ours	**7.827892**	**3.801397**	**4.189983**	**0.245139**
LIO-SAM	15.985649	4.853936	5.585050	0.854267
FAST-LIO	22.136492	8.540459	10.115659	0.788699

3.2.2. GPS Interrupted Experiment

The experiment was conducted with intermittent GPS signal, and its trajectory within the Intelligent Technology Park is shown in Figure 14. Table 4 shows the ATE statistical data of this paper's algorithm and LIO-SAM under GPS interruption conditions.

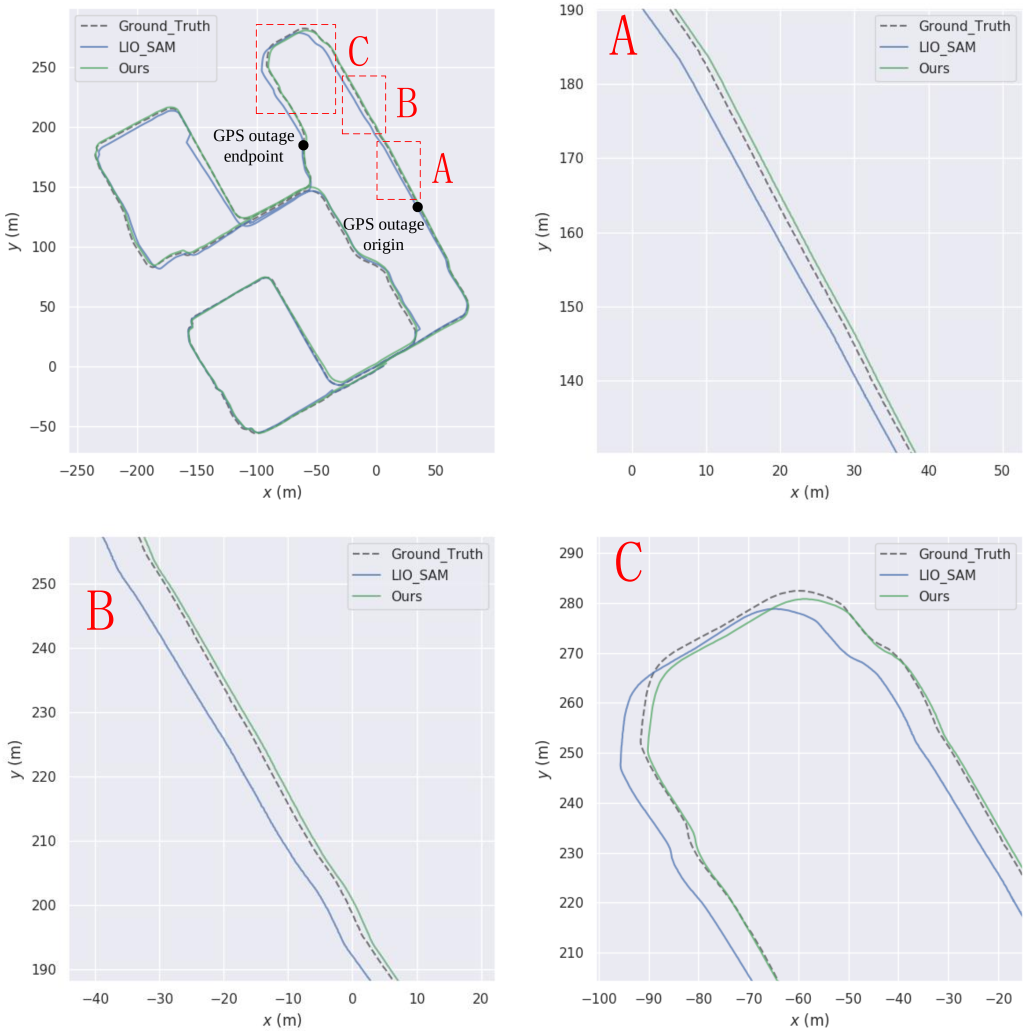


Figure 14. Comparison with LIO-SAM trajectory on our own data set (**A**–**C**) are road sections where our trajectory is better than LIO-SAM.

Despite varying GPS interruption times, the minimum error value remains unchanged, and the maximum error value changes relatively little. This is because the poses associated with the minimum and maximum error values in the entire trajectory of the Intelligent Technology Park are not part of the trajectory produced during GPS interruption. Therefore, the subsequent analysis of ATE statistical indicators will exclude the maximum and minimum error values.

Under the GPS interruption condition for 50 s, this paper's algorithm shows better average error and root-mean-square error compared to LIO-SAM, with improvements of 19.61% and 30.53%, respectively. Under the GPS interruption condition for 100 s, this paper's algorithm demonstrates improvements of 17.33% and 29.11% in terms of average error and root-mean-square error compared to LIO-SAM. Under the GPS interruption condition for 200 s, this paper's algorithm still outperforms LIO-SAM, with improvements of 12.05% and 23.14%, respectively. These results indicate that this paper's algorithm maintains relatively good accuracy improvement even when GPS data is suddenly interrupted for 50 s, and the improvement rate gradually decreases with longer GPS data interruption time. However, the overall trajectory accuracy of this paper's algorithm remains higher than that of LIO-SAM.

Table 4. Comparing the absolute error of our method with LIO-SAM on our own dataset.

Time(s)	Method	Max (m)	Mean (m)	Median (m)	Min (m)
50	Ours	**7.955014**	**4.338442**	**4.700273**	**0.245139**
	LIO-SAM	15.671642	5.396973	6.765650	0.854267
100	Ours	**7.955014**	**4.338442**	**4.700273**	**0.245139**
	LIO-SAM	15.671642	5.396973	6.765650	0.854267
200	Ours	**7.927294**	**4.952018**	**5.397438**	**0.245139**
	LIO-SAM	15.90683	5.630523	7.022866	0.854267

4. Conclusions

This paper presents a SLAM system that combines the IESKF and factor graph approaches. The proposed algorithm tightly integrates the laser rangefinder and IMU to obtain an initial pose estimation using IESKF in the front-end. In the back-end, it fuses front-end odometry, IMU, GPS, and loop closure factors, using a factor graph to achieve precise pose estimation. Additionally, a filtering strategy based on GPS status and confidence is applied to remove abnormal GPS data. Multiple experiments were conducted on the KITTI dataset and our own dataset to verify the accuracy of the SLAM system in scenarios with GPS signal interruption. The results demonstrate that our method outperforms LIO-SAM and FAST-LIO in terms of accuracy when GPS data is interrupted.

Author Contributions: Conceptualization, B.Y. and G.Z.; methodology, B.Y. and G.Z.; validation, G.Z.; data analysis, G.Z.; writing—original draft preparation, G.Z.; writing—review and editing, B.Y., J.L., C.C., E.M. and G.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Heilongjiang Provincial Natural Science Fund Joint Guidance Project, project approval No. LH2020E088, Open Fund Project of State Key Laboratory of Robotics and System, grant number SKLRS-2022-KF-18, National Natural Science Foundation of China, grant number 91948202, National Natural Science Foundation of China, grant number 52175012, and the Fundamental Research Foundation for Universities of Heilongjiang Province, grant number 2022-KYYWF-0122.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Li, J.; You, B.; Ding, L.; Yu, X.; Li, W.; Zhang, T.; Gao, H. Dual-Master/Single-Slave Haptic Teleoperation System for Semiautonomous Bilateral Control of Hexapod Robot Subject to Deformable Rough Terrain. *IEEE Trans. Syst. Man Cybern. Syst.* **2022**, *52*, 2435–2449. [CrossRef]
2. Qin, T.; Li, P.; Shen, S. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Trans. Robots* **2018**, *34*, 1004–1020. [CrossRef]
3. Li, X.; Zhang, W.; Zhang, K.; Zhang, Q.; Li, X.; Jiang, Z.; Ren, X.; Yuan, Y. GPS satellite differential code bias estimation with current eleven low earth orbit satellites. *J. Geod.* **2021**, *95*, 76. [CrossRef]

4. You, B.; Li, J.; Ding, L.; Xu, J.; Li, W.; Li, K.; Gao, H. Semi-Autonomous Bilateral Teleoperation of Hexapod Robot Based on Haptic Force Feedback. *J. Intell. Robot. Syst.* **2018**, *91*, 583–602. [CrossRef]
5. Schmid, K.; Lutz, P.; Tomíc, T.; Mair, E.; Hirschmüller, H. Autonomous vision-based microair vehicle for indoor and outdoor navigation. *J. Field Robots* **2014**, *31*, 537–570. [CrossRef]
6. Gao, Y.; Liu, S.; Atia, M.M.; Noureldin, A. INS/GPS/LiDAR Integrated Navigation System for Urban and Indoor Environments Using Hybrid Scan Matching Algorithm. *Sensors* **2015**, *15*, 23286–23302. [CrossRef]
7. Shamsudin, A.U.; Ohno, K.; Hamada, R.; Kojima, S.; Westfechtel, T.; Suzuki, T.; Okada, Y.; Tadokoro, S.; Fujita, J.; Amano, H. Consistent map building in petrochemical complexes for firefighter robots using SLAM based on GPS and LIDAR. *Robomech. J.* **2018**, *5*, 7. [CrossRef]
8. Abdelaziz, N.; El-Rabbany, A. An Integrated INS/LiDAR SLAM Navigation System for GNSS-Challenging Environments. *Sensors* **2022**, *22*, 4327. [CrossRef]
9. Aboutaleb, A.; El-Wakeel, A.S.; Elghamrawy, H.; Noureldin, A. LiDAR/RISS/GNSS dynamic integration for land vehicle robust positioning in challenging GNSS environments. *Remote Sens.* **2020**, *12*, 2323. [CrossRef]
10. Zhao, S.; Fang, Z.; Li, H.; Scherer, S. A Robust Laser-Inertial Odometry and Mapping Method for Large-Scale Highway Environments. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 3–8 November 2019; pp. 1285–1292. [CrossRef]
11. Qin, C.; Ye, H.; Pranata, C.E.; Han, J.; Zhang, S.; Liu, M. LINS: A Lidar-Inertial State Estimator for Robust and Efficient Navigation. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; pp. 8899–8906. [CrossRef]
12. Xu, W.; Zhang, F. Fast-LIO: A fast, robust LiDAR-inertial odometry package by tightly-coupled iterated Kalman filter. *IEEE Robots Autom. Lett.* **2021**, *6*, 3317–3324. [CrossRef]
13. Xu, W.; Cai, Y.; He, D.; Lin, J.; Zhang, F. Fast-lio2: Fast direct lidar-inertial odometry. *IEEE Trans. Robots* **2022**, *38*, 2053–2073. [CrossRef]
14. Li, J.; You, B.; Ding, L.; Xu, J.; Li, W.; Chen, H.; Gao, H. A Novel Bilateral Haptic Teleoperation Approach for Hexapod Robot Walking and Manipulating with Legs. *Robot. Auton. Syst.* **2018**, *108*, 1–12. [CrossRef]
15. Bai, C.; Xiao, T.; Chen, Y.; Wang, H.; Zhang, F.; Gao, X. Faster-LIO: Lightweight Tightly Coupled Lidar-Inertial Odometry Using Parallel Sparse Incremental Voxels. *IEEE Robots Autom. Lett.* **2022**, *7*, 4861–4868. [CrossRef]
16. Chang, L.; Niu, X.; Liu, T.; Tang, J.; Qian, C. GNSS/INS/LiDAR-SLAM Integrated Navigation System Based on Graph Optimization. *Remote Sens.* **2019**, *11*, 1009. [CrossRef]
17. Kukko, A.; Kaijaluoto, R.; Kaartinen, H.; Lehtola, V.V.; Jaakkola, A.; Hyyppä, J. Graph SLAM correction for single scanner MLS forest data under boreal forest canopy. *Isprs J. Photogramm. Remote Sens.* **2017**, *132*, 199–209. [CrossRef]
18. Hess, W.; Kohler, D.; Rapp, H.; Andor, D. Real-time loop closure in 2D LIDAR SLAM. In Proceedings of the IEEE International Conference Robotics and Automation, Stockholm, Sweden, 16–21 May 2016; pp. 1271–1278. [CrossRef]
19. Chang, L.; Niu, X.; Liu, T. GNSS/IMU/ODO/LiDAR-SLAM Integrated Navigation System Using IMU/ODO Pre-Integration. *Sensors* **2020**, *20*, 4702. [CrossRef]
20. Pierzchała, M.; Giguère, P.; Astrup, R. Mapping forests using an unmanned ground vehicle with 3D LiDAR and graph-SLAM. *Comput. Electron. Agric.* **2018**, *145*, 217–225. [CrossRef]
21. Google Cartographer. Available online: https://google-cartographer.readthedocs.io/en/latest/ (accessed on 9 July 2018).
22. Zhang, J.; Singh, S. Low-drift and Real-time Lidar Odometry and Mapping. *Auton. Robot.* **2017**, *41*, 401–416. [CrossRef]
23. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets robotics: The KITTI dataset. *Int. J. Robots Res.* **2013**, *32*, 1231–1237. [CrossRef]
24. Xue, G.; Wei, J.; Li, R.; Cheng, J. LeGO-LOAM-SC: An Improved Simultaneous Localization and Mapping Method Fusing LeGO-LOAM and Scan Context for Underground Coalmine. *Sensors* **2022**, *22*, 520. [CrossRef]
25. 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 Robots Autom. Lett.* **2021**, *6*, 2068–2075. [CrossRef]
26. Wang, H.; Wang, C.; Chen, C.L.; Xie, L. F-LOAM: Fast LiDAR odometry and mapping. In Proceedings of the 2021 IEEE/RSJ International Conference Intelligent Robots and Systems (IROS) IEEE, Prague, Czech Republic, 27 September–1 October 2021; pp. 4390–4396. [CrossRef]
27. 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 Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 24 October 2020–24 January 2021; pp. 5135–5142. [CrossRef]
28. Li, T.; Pei, L.; Xiang, Y.; Wu, Q.; Xia, S.; Tao, L.; Guan, X.; Yu, W. P3-LOAM: PPP/LiDAR loosely coupled SLAM with accurate covariance estimation and robust RAIM in urban canyon environment. *IEEE Sensors J.* **2021**, *21*, 6660–6671. [CrossRef]
29. Kaess, M.; Johannsson, H.; Roberts, R.; Ila, V.; Leonard, J.; Dellaert, F. iSAM2: Incremental smoothing and mapping with fluid relinearization and incremental variable reordering. In Proceedings of the 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; pp. 3281–3288. [CrossRef]

30. He, J.; Sun, C.; Zhang, B.; Wang, P. Adaptive error-state Kalman filter for attitude determination on a moving platform. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 9513110. [CrossRef]
31. Evo: Python Package for the Evaluation of Odometry and SLAM. Available online: https://github.com/MichaelGrupp/evo (accessed on 5 June 2021).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.