remote sensing

MDPI

Article

Enhanced Strapdown Inertial Navigation System (SINS)/LiDAR Tightly Integrated Simultaneous Localization and Mapping (SLAM) for Urban Structural Feature Weaken Occasions in Vehicular Platform

Xu Xu, Lianwu Guan *, Yanbin Gao, Yufei Chen and Zhejun Liu

College of Intelligent Systems Science and Engineering, Harbin Engineering University, Harbin 150001, China; xuxu66@hrbeu.edu.cn (X.X.)

* Correspondence: guanlianwu@hrbeu.edu.cn

Abstract: LiDAR-based simultaneous localization and mapping (SLAM) offer robustness against illumination changes, but the inherent sparsity of LiDAR point clouds poses challenges for continuous tracking and navigation, especially in feature-deprived scenarios. This paper proposes a novel LiDAR/SINS tightly integrated SLAM algorithm designed to address the localization challenges in urban environments characterized in sparse structural features. Firstly, the method extracts edge points from the LiDAR point cloud using a traditional segmentation method and clusters them to form distinctive edge lines. Then, a rotation-invariant feature—line distance—is calculated based on the edge line properties that were inspired by the traditional tightly integrated navigation system. This line distance is utilized as the observation in a Kalman filter that is integrated into a tightly coupled LiDAR/SINS system. This system tracks the same edge lines across multiple frames for filtering and correction instead of tracking points or LiDAR odometry results. Meanwhile, for loop closure, the method modifies the common SCANCONTEXT algorithm by designating all bins that do not reach the maximum height as special loop keys, which reduce false matches. Finally, the experimental validation conducted in urban environments with sparse structural features demonstrated a 17% improvement in positioning accuracy when compared to the conventional point-based methods.

Keywords: 3D LiDAR navigation; SLAM; tightly integrated navigation; LiDAR odometry and mapping; urban structural feature weaken occasions

Citation: Xu, X.; Guan, L.; Gao, Y.; Chen, Y.; Liu, Z. Enhanced Strapdown Inertial Navigation System (SINS)/LiDAR Tightly Integrated Simultaneous Localization and Mapping (SLAM) for Urban Structural Feature Weaken Occasions in Vehicular Platform. *Remote Sens.* **2024**, *16*, 2527. https://doi.org/10.3390/rs16142527

Academic Editors: Wanshou Jiang, San Jiang, Duojie Weng and Jianchen Liu

Received: 29 April 2024
Revised: 5 July 2024
Accepted: 7 July 2024
Published: 10 July 2024

1. Introduction

In recent years, the domain of simultaneous localization and mapping (SLAM) [1] has been an integral part of autonomous navigation, especially in environments where the reception of global navigation satellite system (GNSS) signals is unreliable or absent and where dynamic environmental conditions are the norm. SLAM aims to determine a robot's pose while simultaneously generating a map of its environment using onboard sensors. This process occurs in environments that may be unknown or partially known. The diversity of applicable sensors in use has naturally led to the bifurcation of SLAM into two primary SLAM categories: LiDAR-based SLAM and visual SLAM. Visual SLAM encompasses various subtypes, including monocular, stereo, and RGB-D [2]. LiDAR-based approaches demonstrate superior accuracy in pose estimation and maintain robust performance across varying environmental conditions, such as time of day and weather. In contrast, visual SLAM, as illustrated in Figure 1, is highly susceptible to factors like lighting and the availability of distinctive features, thus potentially limiting its effectiveness in certain settings [3]. Therefore, this paper concentrates on navigation systems leveraging LiDAR technology.

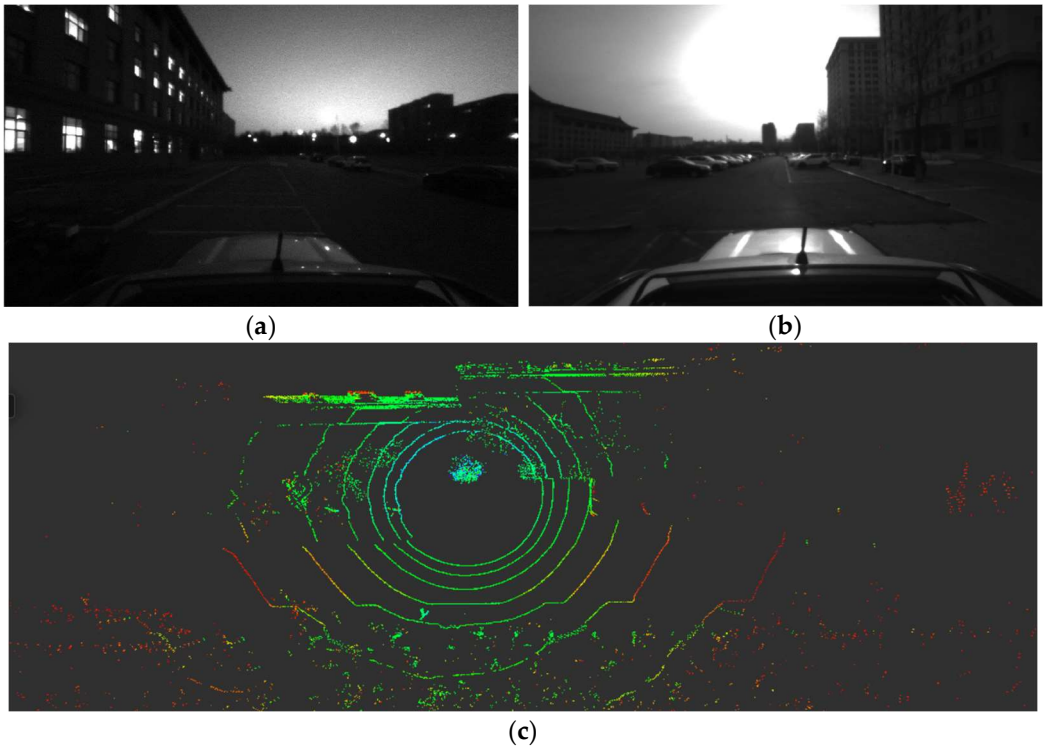


Figure 1. The figures illustrate the impact of the environment on different sensors; (**a**,**b**) show the effect of darkness and illumination on the visual sensor, respectively, while (**c**) indicates that LiDAR can work normally under such conditions.

The last two decades have witnessed significant strides in LiDAR-based SLAM methodologies, fueled by advancements in computer processing power and optimization algorithms. While machine learning-based LiDAR SLAM methods are constrained by the scope and quality of training data, two primary categories dominate the current landscape: normal distributions transform (NDT) [4] and iterative closest point (ICP) [5] algorithm. The NDT approach, which has seen extensive application in 2D LiDAR SLAM scenarios, entails the discretization of the point cloud data into a grid-like structure, the computation of Gaussian distributions for each cell, and subsequent alignment-based matching and fitting procedures [4]. However, the transition from 2D to 3D applications has exponentially increased computational demands, posing a significant challenge in meeting the stringent requirements for real-time processing [6]. To put it simply, when classified solely based on the quantity of point clouds and meshes, the computational data of the most basic 3D LiDAR are at least 16 times that of a 2D LiDAR, as they have at least 16 scanning projection planes. Although efforts to mitigate this issue have been made through algorithms such as SEO-NDT [7] and KD2D-NDT [8], they have occasionally resulted in trade-offs concerning accuracy and processing time in certain scenarios. ICP-based methods face similar challenges. However, the advent of LiDAR odometry and mapping (LOAM) [9] has marked a pivotal shift in focus; LOAM focuses the iteration process towards feature-rich points rather than the entire point cloud. This paradigm shift has propelled the widespread implementation of LOAM-inspired ICP techniques in addressing LiDAR SLAM challenges over the past decades.

LOAM distinguishes itself from conventional ICP techniques by classifying points in the point cloud based on their smoothness. This involves identifying and extracting "edge points", which are characterized by coarse texture, and "planar points", which exhibit fine texture. Subsequently, the derived feature points are systematically selected through a sector-based averaging technique. The system leverages these refined point clouds; the system performs odometry calculations at a frequency of 10 Hz using LiDAR data. Following the odometry computation, the aggregated point clouds are then employed for

mapping at a reduced frequency of 1 Hz, thereby achieving a more accurate and efficient representation of the environment. To address LOAM's limitations in computational demands and loop closure, Shan and Englot proposed LeGO-LOAM [10]. This method, which stands for lightweight and ground-optimized LiDAR odometry and mapping, is specifically tailored for real-time six-degrees-of-freedom pose estimation with ground vehicles. However, further experiments have shown that the strategy of entirely segregating ground points from the surrounding point cloud environment for separate matching can result in a notable vertical drift. Furthermore, the methodologies for loop closure still face certain challenges.

Li He and colleagues investigated the application of Multiview 2D projection (M2DP) [11] to describe 3D points to achieve loop closure, but their findings showed limited scope and efficacy. Scan Context [12] and its advanced iteration, Scan Context++ [13], were introduced by Giseop Kim in 2018 and 2021, respectively. These innovative approaches have rapidly gained recognition as leading solutions for loop closure in 3D LiDAR-based SLAM. This is a non-histogram-based global descriptor that directly captures egocentric structural information from the sensor's field of view without relying on prior training. However, the aforementioned methods and their derivatives, such as F-LOAM [14], do not utilize strapdown inertial navigation systems (SINSs) or only use them for the rectification of LiDAR point clouds.

Compared to the mature field of vision-aided SINS, the integration of SINS and LiDAR within LiDAR-based SLAM algorithms remains largely unexplored. A study [15] employed a loosely coupled extended Kalman filter (EKF) to fuse IMU and LiDAR data within a two-dimensional framework. However, this approach lacked the robustness to handle the complexities of three-dimensional or multifaceted environments. Furthermore, a scholarly review published in 2022 [16] emphasized that within the majority of current systems employing the SINS/LiDAR integration systems, the SINS primarily functions to smooth trajectories and mitigate distortions. IMU data are often optionally integrated to predict platform motion and enhance registration accuracy during abrupt maneuvers. However, only gyroscopic measurements between consecutive LiDAR scans are utilized. Although these studies and related works often self-identify as "loose integration" based on the data fusion strategies outlined in this article, a more accurate designation would be "pseudo integration".

As illustrated in Figure 2, the concept of loose integration in LiDAR/SINS systems can be redefined from the GNSS/SINS loose integration navigation system. This approach involves combining position and other navigation data obtained from different sensors. In this process, none of the sensors involved in the integration have undergone in-depth data integration, but only a simple integration of the navigation results. By applying this redefined concept of loose integration, it becomes evident that studies such as [17,18], while claiming to employ tight integration, actually align more closely with the characteristics of loose integration. Specifically, these studies treat the individual systems as black boxes, focusing solely on integrating their outputs to generate the final navigation solution rather than performing in-depth data extraction and analysis.

To achieve a deeper level of sensor fusion than loose integration, the integration process should occur before the generation of individual navigation solutions. For instance, in a GNSS/SINS system, this translates to integrating data during the pseudorange measurement stage, prior to GNSS position determination. A key advantage of tightly coupled GNSS/SINS integration [19] over the loosely coupled approach is its reduced reliance on a high number of visible satellites. This integration scheme can function even with a single observable satellite, unlike loose integration, which typically requires at least four. Investigating tight integration within SLAM systems necessitates understanding the nature of the data employed for navigation. In LiDAR-based systems, these data comprise point clouds, while vision-based systems utilize feature point information. Some studies [20,21] have demonstrated that within the SLAM framework, the concept of lines exhibits greater stability than points, particularly during data transformations (rotation and translation) across multiple frames. Similarly, research on multi-frame feature tracking within multi-

state constrained Kalman filters for vision-aided inertial navigation [22] has validated the enhanced accuracy and robustness of this approach compared to traditional methods. This has led to the development of a prototype tightly integrated LiDAR/SINS navigation system that utilizes line features extracted from the LiDAR point cloud as observations. The system employs continuous, multi-frame tracking of these line features to refine the SINS data. However, due to the sparse nature of the LiDAR point cloud, it is difficult to accurately track the same line. Consequently, revisiting the concept of distance as a measurement, akin to its application in GNSS/SINS systems, becomes crucial. Notably, distance, being a scalar quantity, offers a significant advantage—rotational invariance. This property can substantially reduce the computational burden of the integration process. The algorithm's core principle centers on leveraging shared features, specifically line distances, across multiple frames to enhance Kalman filter accuracy. In summary, this paper presents the following contributions:

1. This paper refines the edge point extraction process of the LOAM algorithm by implementing a more granular clustering approach. By classifying clustered edge points as either convex or concave, the mapping precision is enhanced. Leveraging the rotational invariance of line distances, a Kalman filter is developed that employs line distance error as its primary observation metric. This approach improves the system's robustness and accuracy.
2. This paper presents structural modifications to the LOAM algorithm that are predicated on the Scan Context framework to optimize its performance and ensure the data processing occurs more efficiently. The experiments have proven that the situation of incorrect loop closures in LiDAR SLAM has been mitigated effectively.
3. Extensive experiments conducted in various on-campus and off-campus environments validate the proposed algorithm and offer comparisons with traditional methods. These experiments highlight the superior performance of the proposed algorithm.

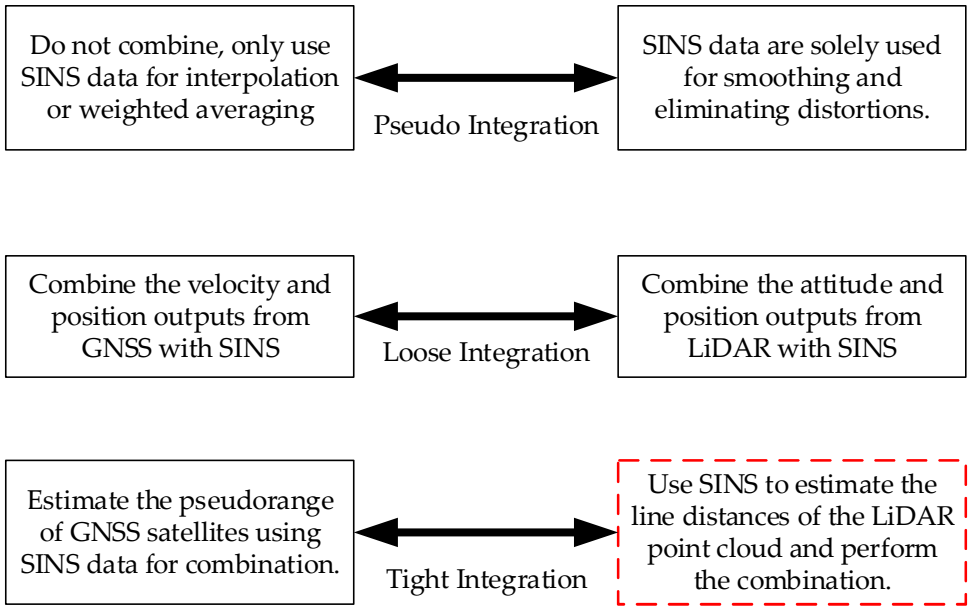


Figure 2. Use the concept of traditional SINS/GNSS integrated navigation systems to redefine the LiDAR integrated navigation system.

2. Method

This section outlines the workflow of our algorithm. This includes an insightful overview of the foundational principles that govern the pertinent hardware components, coupled with a thorough elucidation of the methodologies employed for the preprocessing of data. Section 3 will then delve into the system error model and measurement model, providing a comprehensive analysis of these crucial framework elements. Additionally, to

elucidate the algorithmic details, the subsequent sections of this paper will operate under the assumption that the LiDAR point cloud was sourced from a 16-line LiDAR system by default. This is representative of commonly used systems such as Velodyne's VLP-16 (Velodyne Acoustics GmbH, Hamburg, Germany) and the LeiShen MS-C16 [23] (Leishen Intelligent System Co., Ltd., located in Shenzhen, China) employed in the experiments of this paper. These systems, with a horizontal angular resolution of 0.2° and a vertical resolution of 2°, generate a range image of 1800 by 16 pixels [23]. This translates to a point cloud with 16 projection planes, each containing 1800 points.

2.1. Algorithm Overview

Figure 3 provides the overview of a tightly integrated LiDAR/SINS SLAM algorithm. Let P be the original points received in a laser scan. However, because scanning occurs over a timeframe t (typically exceeding 0.1 s), the resulting point cloud represents the environment over this duration rather than instantaneously. Consequently, in dynamic environments, the recorded point cloud may exhibit distortions caused by movements, particularly pronounced during significant angular variations. To mitigate this, it is essential to utilize the high-frequency motion data provided by the SINS to project all points onto the reference timestamps, either the beginning of the period t_{k-1} or the end t_k.

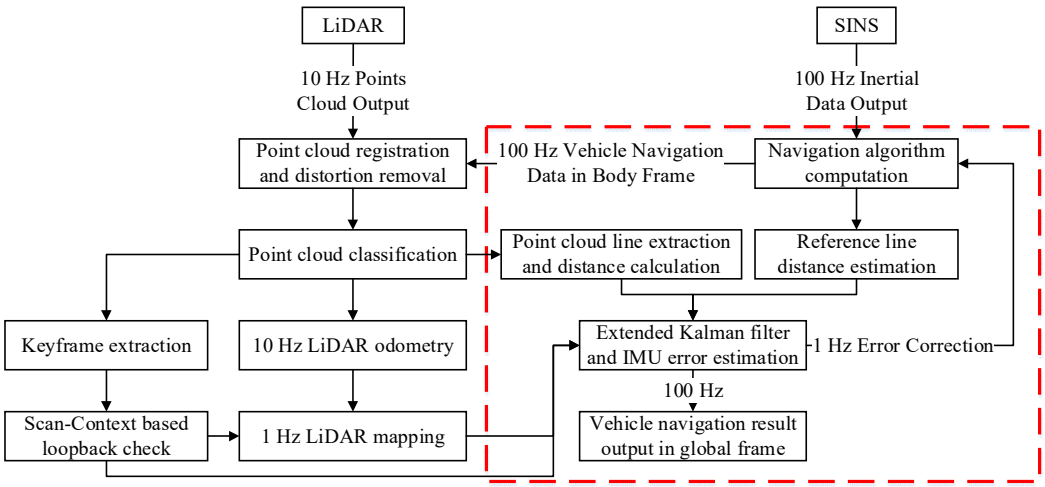


Figure 3. The algorithm overview of SINS-based 3D LiDAR tightly integrated SLAM.

After processing, the point cloud is denoted as $\hat{P}$ and proceeds to the next stage. Here, each point undergoes a meticulous classification into four categories based on its properties: (a) ground points, representing the surface on which the vehicle travels; (b) planar points, indicative of flat surfaces except the ground; (c) edge points, which demarcate boundaries or the perimeter of objects; and (d) the others, encompassing all points that do not fit into the previous categories. The subsequent section will elaborate on the point cloud classification technique, ensuring a thorough understanding. All points outside the ground in a key frame are compressed into scan context descriptors, and the key frames are set based on distance and the structure of the point clouds. Concurrently, after the clustering process, edge points are re-extracted to form edge lines. These edge lines will then serve as a basis for further computation of the reference line distances and facilitate tightly coupled filtering.

The LiDAR/SINS odometry primarily relies on the SINS navigation results, and the outputs further processed by LiDAR mapping, which matches and registers the undistorted point cloud onto a map at a frequency of 1 Hz. The Scan Context system performs loop closure detection based on both time and the distance traveled. When the similarity measure in the loop closure detection reaches a certain threshold, it is considered that the vehicle has returned to a previously visited location. Subsequently, the system optimizes

the overall trajectory using this information. Successful loop closure detections will also contribute to the refinement of the SINS navigation and Kalman filtering processes.

2.2. Point Cloud Classification and Point Cloud Lines Extraction

Figure 4 shows the undistorted raw point cloud, ground points, edge points, and planar points, as well as edge line points, respectively. The following will detail the extraction methods for each point type.

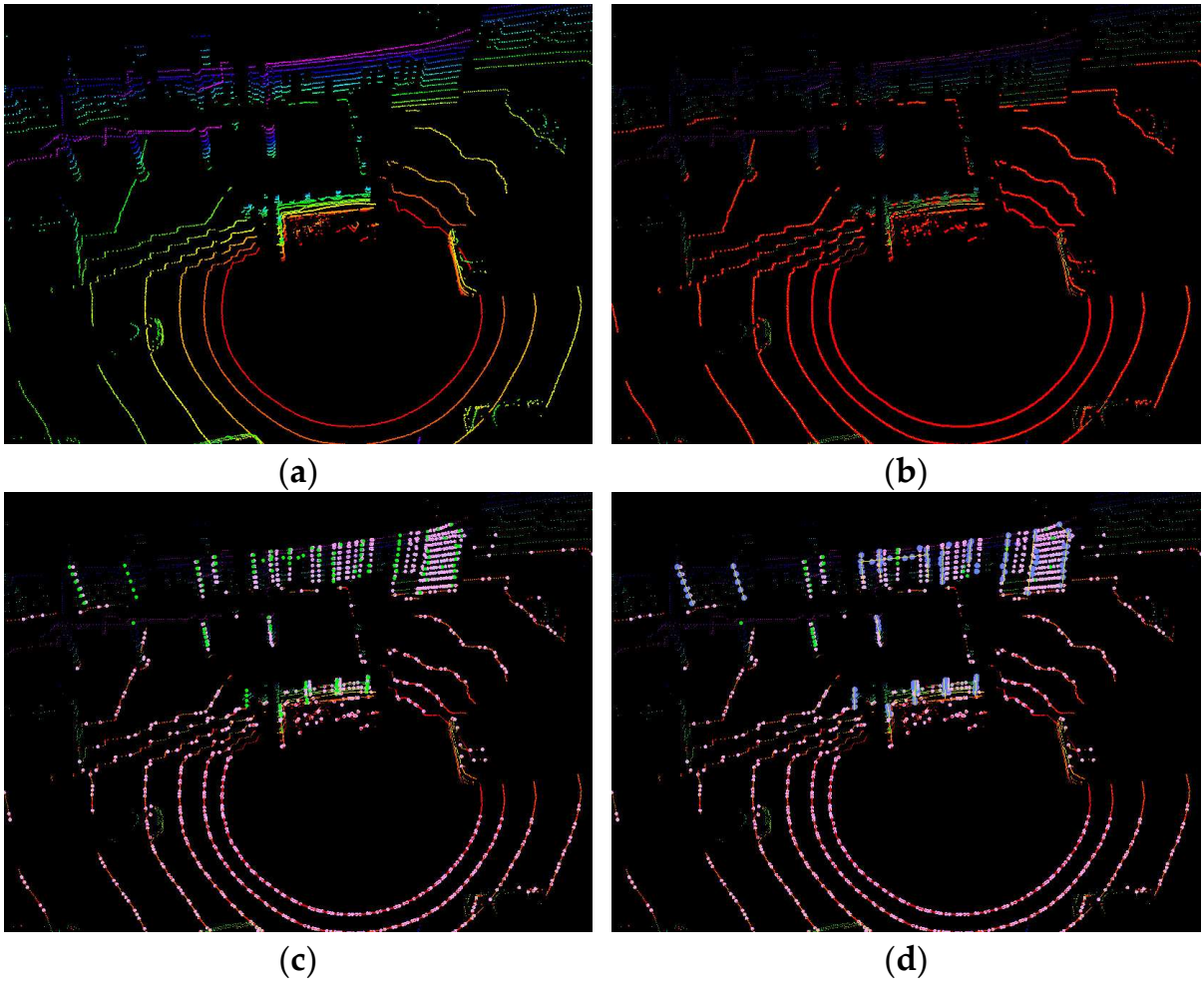


Figure 4. Feature extraction process for a scan in noisy environment. The original point cloud is shown in (**a**). (**b**) The red points are labeled as ground points. The rest of the points are the points that remain after segmentation. This method will be shown in Section 2.2.1. (**c**) Green and pink points indicate edge and planar features, which are mentioned in Section 2.2.2. (**d**) The blue points represent edge line points. The specific extraction method is explained in Section 2.2.3.

2.2.1. Ground Points

LeGO-LOAM employs a straightforward and efficient ground point extraction method, which involves specifically examining the 8 lines out of the total 16 that are positioned below 0° for detection [23].

In point cloud $\hat{P}$, point clouds are labeled with rings and scan sequence; let $p_{i,j} \in \hat{P}$, $i = 1, 2, 3 \cdots 16$, *and* $j = 1, 2, 3 \cdots 1800$. As shown in Figure 5, to calculate the angle between adjacent points $p_{i,j}$ and $p_{i+1,j}$, this paper assumes their coordinate differences are denoted as $diffx$, $diffy$ and $diffz$. The angle θ could be set as:

$$\theta = \tan^{-1}\left(diffz, \sqrt{diffx^2 + diffy^2}\right) \tag{1}$$

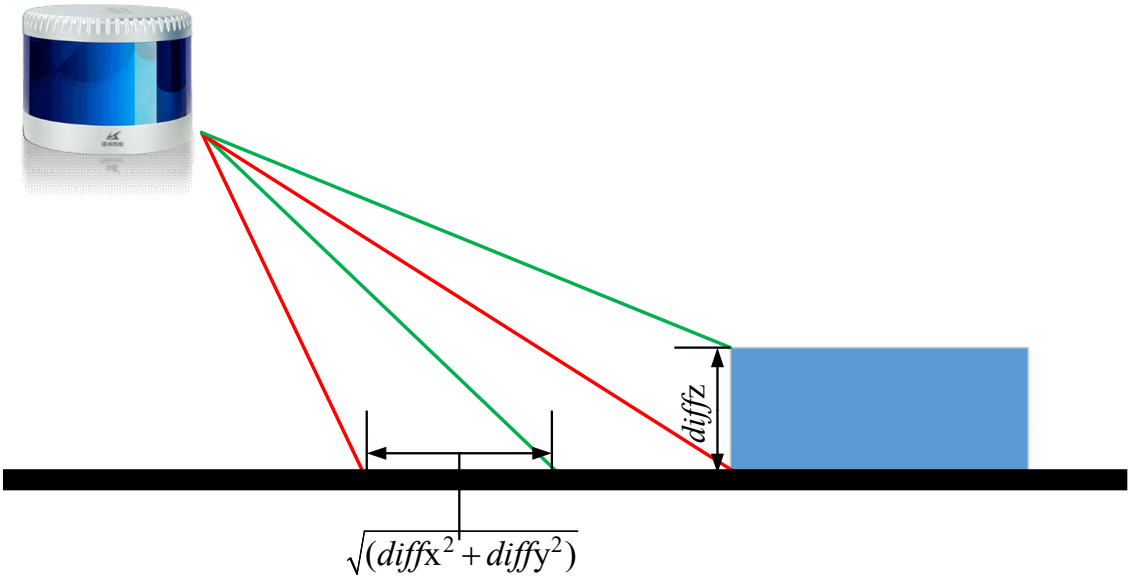


Figure 5. Ground points extracted from the point cloud $\hat{P}$.

Once $\theta < 10°$, points are marked as candidate ground points. Furthermore, an advanced point cloud sieving process [24] will utilize the RANSAC (random sample consensus) [25] technique to confirm the identification of ground points. This step is critical to avoid the misclassification of non-ground points as ground points, thereby ensuring the accuracy and reliability of the ground detection process. The fitted ground equation is as follows:

$$Ax + By + Cz + d = 0 \tag{2}$$

Then, an image-based segmentation method [26] is applied to the range image to group points into many clusters. Points from the same cluster are assigned a unique label.

2.2.2. Edge and Planar Points

The feature extraction process is similar to the method used in [9]; but, instead of extracting from the raw point cloud $\hat{P}$, we exclusively utilize the portion of the point cloud that remains unmarked as ground points. Let S be the set of points of p_i from the same ring of the point clouds. Half of the points are on either side of p_i. The set for this paper is presented in Table A1. Using the range values computed during segmentation, we can evaluate the roughness of point p_i in S,

$$c = \frac{1}{|S| \cdot \|r_i\|} \left\| \sum_{j \in S, j \neq i} (r_j - r_i) \right\| \tag{3}$$

where r_j means the range from p_i to the center of LiDAR.

Similar to LOAM, we use a threshold c_{th} to distinguish different types of features. We call the points with c larger than c_{th} edge points, and the points with c smaller than c_{th} planar points. Then, we sort the edge and planar points from minimum to maximum. The point cloud is segmented into several distinct parts, and a specific number of feature points are extracted from within each segment.

Following the extraction of feature points, another attribute will be computed, specifically, the concavity or convexity of the edge points. Figure 6 shows the difference between the concave points and convex points. Compare the distances between a specific point p_i and the remaining points within set S. If the majority of these points have distances greater than that of p_i, then p_i is considered a convex point. Otherwise, it is a concave point.

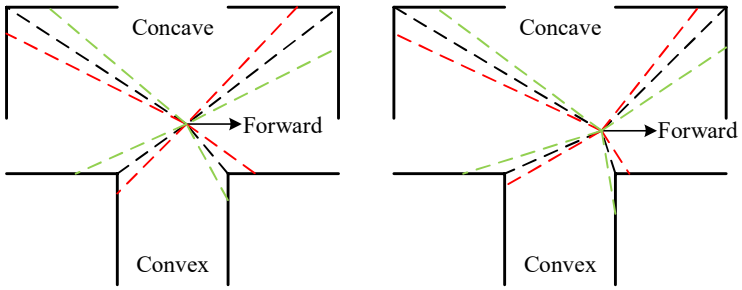


Figure 6. The concavity or convexity of the edge points.

2.2.3. Edge Lines

After classifying edge points as concave or convex, this paper employs K-means clustering [27] to group them into lines. Similarly, these lines will inherit concavity or convexity from the points that constitute them. This choice of using edge lines instead of individual edge points for subsequent computations stems from the inherent sparsity of LiDAR point clouds. Ensuring the capture of the exact same point across consecutive scans is a challenging proposition. In contrast, lines, when considered as collective entities, offer a higher degree of continuity and are much more amenable to persistent tracking. This approach enhances the reliability and robustness of the subsequent processing steps.

Section 3 will elaborate on the method for calculating point-to-line distances and the line selection criteria.

2.3. Scan Context

Scan Context was inspired by Shape Context [28], proposed by Belongie et al.; it is an algorithm for place recognition using 3D LiDAR scans. It works by:

1. Partitioning the point cloud into bins based on azimuthal and radial directions.
2. Encoding the point cloud into a matrix where each bin's value is the maximum height of points within it.
3. Calculating similarity between scan contexts using a column-wise distance measure.
4. Employing a two-phase search for loop detection that is invariant to viewpoint changes.

Figure 7 shows the bin division along azimuthal and radial directions. Using the top view of a point cloud from a 3D scan, the paper [10] partitioned ground areas into bins, which were split according to both azimuthal (from 0 to 2π within a LiDAR frame) and radial (from center to maximum sensing range) directions. They referred to the yellow area as a ring, the cyan area as a sector, and the black-filled area as a bin.

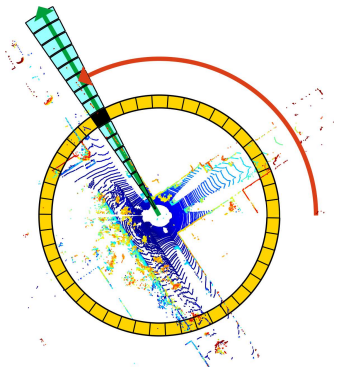


Figure 7. The scan context bins.

However, assigning the maximum height of points within a bin a value in the scan context can be problematic in certain situations. As depicted in Figure 8, due to the formation principle of LiDAR point clouds, the point cloud does not fully unfold at close ranges,

which may result in the highest point not accurately representing the actual environmental point cloud.

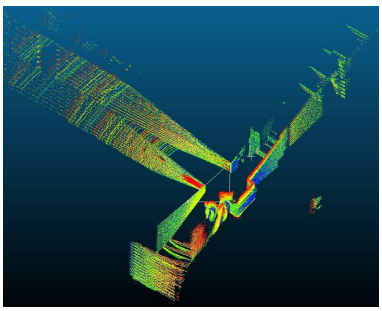


Figure 8. Close-range point cloud scanning scenario in Cloud-Compare software (2.13).

A straightforward and effective solution is to perform a ring-based search for the highest point. If the highest point lies within the outermost ring of the 3D LiDAR and is lower than the adjacent bins, an additional annotation is made to record that the highest point has not been detected. The marked bin can then serve as a ring-key in scan context for the initial match.

Simultaneously, because the point cloud distribution is dense near and sparse far, for each point cloud P selected as a key frame, we can first calculate its centroid:

$$\hat{P}(O) = \frac{1}{n}\sum p_{i,j},\ \overline{p_{i,j}} = p_{i,j} - \hat{P}(O) \tag{4}$$

where n is the total number of the point cloud and $\overline{p_{i,j}}$ is the point cloud $p_{i,j}$ transformed back to the center of the original point cloud.

As is shown in Figure 9, the transformed point cloud will have a common center, which will save a significant amount of time in subsequent scan context description and matching processes.

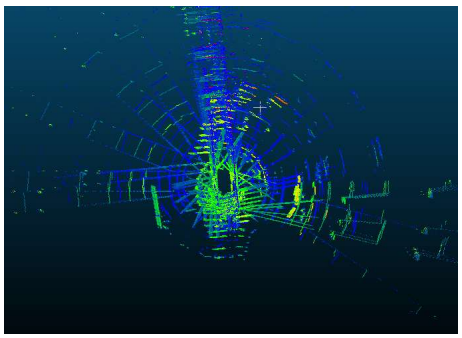


Figure 9. Transformed point clouds stacked in Cloud-Compare software (2.13).

3. LiDAR/SINS System Model

3.1. System Error Model

The SINS integrated navigation system error model was designed following the list in [29]:

$$\delta\dot{x} = F\delta x + Gw \tag{5}$$

$$F = \begin{bmatrix} F_{11} & F_{12} & 0_{3*3} & 0_{3*3} & 0_{3*3} \\ F_{21} & F_{22} & F_{23} & 0_{3*3} & R_b^n \\ F_{31} & F_{32} & F_{33} & R_b^n & 0_{3*3} \\ 0_{3*3} & 0_{3*3} & 0_{3*3} & F_{44} & 0_{3*3} \\ 0_{3*3} & 0_{3*3} & 0_{3*3} & 0_{3*3} & F_{55} \end{bmatrix} \tag{6}$$

The specific parameters in Equation (6) can be referred to in Equation (A1). And ϕ and λ are local latitude and longitude. R_M and R_N are meridian radius and normal

radius. The h is the geodetic height. R_b^n is the transformation matrix from body frame to navigation frame.

For system state variables, $\delta x = \left[\delta p^n, \delta v^n, \delta a^n, b_\omega^n, b_f^n\right]^T$, δp^n denotes the positional errors in longitude, latitude, and height. δv^n presents the velocity errors related to the three directions above. δa^n is the error attitude. b_ω^n and b_f^n are the sensor noise errors of the gyroscopes and accelerometers, respectively.

The w is system noise, and the corresponding system noise matrix is given by:

$$G = \left[0_{9*1} \quad \sqrt{2\beta_{\omega x}\sigma_{\omega x}^2} \quad \sqrt{2\beta_{\omega y}\sigma_{\omega y}^2} \quad \sqrt{2\beta_{\omega z}\sigma_{\omega z}^2} \quad \sqrt{2\beta_{fx}\sigma_{fx}^2} \quad \sqrt{2\beta_{fy}\sigma_{fy}^2} \quad \sqrt{2\beta_{fz}\sigma_{fz}^2}\right]^T \tag{7}$$

where $\beta_{\omega x}$, $\beta_{\omega y}$, and $\beta_{\omega z}$ denote reciprocals of the correlation times of the autocorrelation sequence of b_ω^n while β_{fx}, β_{fy}, and β_{fz} are related to b_f^n. The $\sigma_{\omega x}^2$, $\sigma_{\omega y}^2$, $\sigma_{\omega z}^2$, σ_{fx}^2, $\sigma_{fy,}^2$ and σ_{fz}^2 are variance associated with gyroscope and accelerometer errors.

3.2. The Observation Model

Traditionally, LiDAR-IMU integration has followed a loosely coupled approach. The observation variables of the model defined as the estimated position errors are:

$$Z = \begin{bmatrix} \phi_{Lidar} - \phi_{SINS} \\ \lambda_{Lidar} - \lambda_{SINS} \\ h_{Lidar} - h_{SINS} \end{bmatrix} \tag{8}$$

However, this integration approach merely treats LiDAR and SINS as two black boxes, simply combining their output results without any deeper level of mutual correction. This paper draws on the concept of tight integration between GNSS and SINS, selecting the error of the reference line distance d as the observation variable for filtering.

The selection of the error of d as the observation variable is based on the following considerations:

1. Frame invariance: In navigation systems, the relative orientation between the body frame and the navigation frame continuously changes. Distance, however, remains consistent across different frames.
2. Robustness to data loss: Compared to vision data, LiDAR point clouds are inherently sparse. Using feature points and their associated information directly as observation variables increases the susceptibility to data loss.

In three-dimensional space, the distance from a point L_0 to a straight line that was built by points L_1 and L_2 can be calculated as follow:

$$d = \frac{|(L_0 - L_1) \times (L_2 - L_1)|}{|L_2 - L_1|} \tag{9}$$

Imagine the points L_1 and L_2 with coordinates $(x_{l1},\ y_{l1}, z_{l1})$ and $(x_{l2},\ y_{l2}, z_{l2})$, respectively, and the origin of the vehicle or robot in time t_k could be estimated as $\widetilde{p_k} = \left(\widetilde{x_k}, \widetilde{y_k}, \widetilde{z_k}\right)$ while the ground truth is $p_k = (x_k, y_k, z_k)$; the relationship between them is:

$$\widetilde{p_k} = \begin{Bmatrix} \widetilde{x_k} \\ \widetilde{y_k} \\ \widetilde{z_k} \end{Bmatrix} = \begin{Bmatrix} x_k + \Delta x_k \\ y_k + \Delta y_k \\ z_k + \Delta z_k \end{Bmatrix} \tag{10}$$

where Δx_k, Δy_k, and Δz_k are the position errors in the ENU directions and $\delta p = (\Delta x_k/(R_{N+h})cos\phi, \Delta y_k/(R_M + h), \Delta z_k)\prime$. The figure above represents the relationship between the vehicle and the reference line in three-dimensional space, where L_0' denotes the actual position of the vehicle and L_0 signifies the vehicle's estimated position.

The measurement parameter in the EKF system is:

$$Z = \Delta d = d' - d \tag{11}$$

As is shown in Figure 10, Δd is much smaller than d or $d\prime$; it can be approximately considered that the normal vector to the line L_1L_2 connecting L_0' and L_0 is consistent. Then, the relationship between Δd and δp could be set as follows:

$$\Delta d = \vec{n}.\delta p \tag{12}$$

where $\vec{n} = (n_x, n_y, n_z)$ is the normalized vector from L_0L_d. Meanwhile, the corresponding system design matrix H is:

$$H = \begin{bmatrix} n_{x1}(R_N + h)cos\phi, n_{y1}(R_M + h), n_{z1}, 0_{1*12} \\ n_{x2}(R_N + h)cos\phi, n_{y2}(R_M + h), n_{z2}, 0_{1*12} \\ n_{x3}(R_N + h)cos\phi, n_{y3}(R_M + h), n_{z3}, 0_{1*12} \\ \cdots \\ n_{xi}(R_N + h)cos\phi, n_{yi}(R_M + h), n_{zi}, 0_{1*12} \end{bmatrix} \tag{13}$$

where i is the number of reference lines selected in the integrated system. It will always be changed with the changing of point clouds.

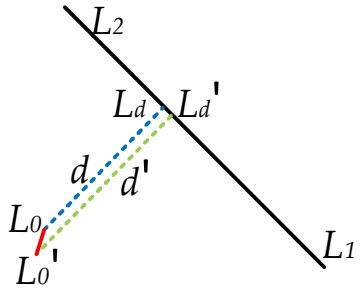


Figure 10. The distance from a point to a line.

For the LiDAR/IMU EKF system, the formulation linking the system observation variables and state variables is:

$$Z = H\delta x + v \tag{14}$$

where Z, H, and δx were defined previously and v represents the observation noise. As mentioned above, the LiDAR point cloud is the covariance value of v. LiDAR point clouds exhibit sparsity, and the uncertainty of a single point affects its entire surrounding space. The surrounding space is a truncated segment of a torus. For ease of calculation, this paper assumes it to be a rectangular prism, and its volume represents the covariance value of the measurement noise v.

For example, imagine a special distance of the reference line d_0. The covariance value of the noise of d_0 is $d_0^2(\sin^2\theta_h + \sin^2\theta_v)$ and θ_h is the horizontal separation angle of the LiDAR device while θ_v is the vertical separation angle.

As mentioned above, i refers to the number of reference lines; in this stated equation, this paper proposes that the following principles should be adhered to in controlling the reference lines involved in the filtering process.

1. Region division: The point cloud is segmented into multiple regions based on the scanning direction. Each region is characterized by the edge lines exhibiting distinct convexity/concavity properties.
2. Reference line tracking: The position of each reference line is tracked across multiple frames using SINS transformations. This ensures the consistent matching of the same reference line over an extended period.
3. Dynamic reference line management: Due to the inherent sparsity of point clouds, reference lines exceeding a predefined distance threshold are discarded. New reference lines are introduced to maintain robust matching.

3.3. Tracking of Reference Lines

1. The selection rules of the reference lines are mentioned in Section 3.2. Here, the tracking of these lines will be revealed with details.
2. As mentioned in Section 2.3, scan context built a series of point bins to extract the point cloud information. Imagine the attitude transformation matrix during the tracking period is R_{3*3} while the t_{3*1} represents the displacement provided by SINS. The projections L_1', L_2' of points L_1 and L_2 in the new point cloud could be calculated in Equation (12):

$$\begin{bmatrix} L_1' \\ 1 \end{bmatrix} = \begin{bmatrix} R_{3*3}^T & -t_{3*1} \\ 0_{1*3} & 1 \end{bmatrix} \begin{bmatrix} L_1 \\ 1 \end{bmatrix} \tag{15}$$

3. Connect L_1 and L_2 to obtain the scan context bins they pass through. By statistically analyzing these bins with their adjacent bins, select the edge line L_{new} that is also located in the same bin area. To further determine whether it is derived from a change in the original edge line, in addition to judging its concavity and convexity as well as the concavity and convexity and distance of the closest edge line, a further similarity analysis can be conducted on the line vectors.

$$s = 1 - \frac{\overrightarrow{L_1'L_2'}.\overrightarrow{L_{new}}}{\left\|\overrightarrow{L_1'L_2'}\right\|\left\|\overrightarrow{L_{new}}\right\|} \tag{16}$$

4. The smaller the value of s is, the higher the similarity between the two lines is. Based on the above conditions, it can be determined whether the new edge line is the target that needs to be tracked. The final threshold selection for this paper can be referred to in the data presented in Table 1.

Table 1. Parameters of method in this paper.

Parameters	Value
Separated Point Cloud Region	8
Reference Line Distance Threshold	80 m
Set of Points	10
Edge Line Similarity Detection Threshold	0.01

4. Experiment

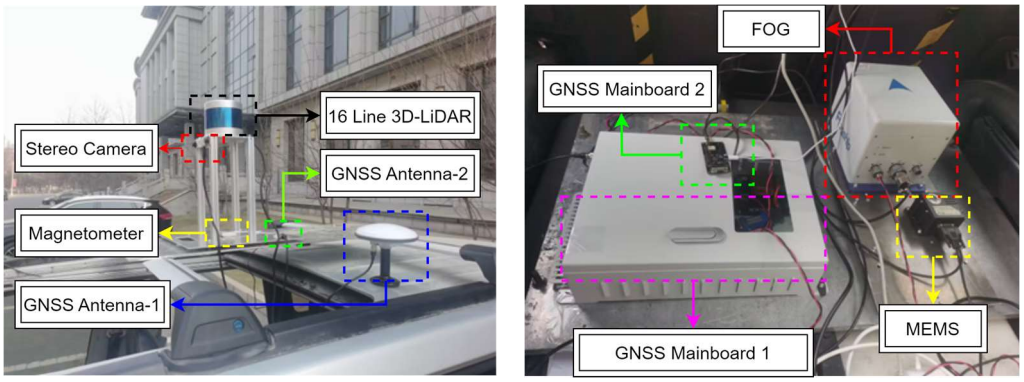
4.1. Algorithm Parameter Settings

As mentioned in the above text, the algorithm parameter settings for the relevant experiments of this paper are all listed in the Table 1.

4.2. Sensors System

All the sensors were mounted on a sport utility vehicle (SUV) for data collection. The LiDAR unit and the GNSS antennas were installed on the roof, while the SINS equipment and power supply were secured within the SUV.

Figure 11 depicts the experimental setup, which utilized a high-performance fiber-optic gyroscope navigation system (Self-developed experimental equipment of Harbin Engineering University, Heilongjiang, China.) integrated with a GNSS receiver (K823 GNSS receiver, ComNav Technology Ltd., Shanghai, China) to provide ground truth data. Table 2 summarizes the specifications of this system. The precise alignment of sensor positions is crucial to minimize navigation errors arising from lever arm effects. Therefore, the central positions of all sensors were carefully measured and aligned with the azimuth axis. Detailed measurement data are available in Table A1. Note that this table omits sensors with less stringent relative positioning requirements, such as the magnetometer used for initial SLAM orientation.



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

Figure 11. The experimental hardware: (**a**) shows the 3D LiDAR and GNSS antenna while (**b**) shows the GNSS processing, fiber optic gyroscope, and MEMS.

Table 2. The specifications of reference integrated navigation system.

Reference Accuracy	Specifications
Pitch	<0.02°
Yaw	<0.02°
Heading	<0.05°
Velocity (Integrated)	<0.1 m/s
Positioning (Integrated)	<1 m
Output Rate	100 Hz

The LiDAR sensor employed in this study was the Leishen 16 Line 3D-LiDAR, with performance parameters detailed in Table 3.

Table 3. The performance parameters of 3D-LiDAR.

Performance	Parameters
Detection Range	200 m
Point Rate	320,000 pts/s (single echo)
Distance Measurement Accuracy	±3 cm
Laser Wavelength	905 nm
Maximum Echo Count for Reception	2
Scanning Channels	16
Field-of-View Angle	360° × −15°~15°
Scanning Frequency	5~20 Hz
Angular Resolution	5 Hz: 0.09°/10 Hz: 0.18°
Power Supply Range	9 V~36 V DC
Operating Temperature	−20 °C~55 °C

Table 4 lists the performance parameters of the ADIS16445 (Analog Devices, Inc., Wilmington, MA, USA) Micro-Electro-Mechanical System (MEMS), a complete inertial system comprising a tri-axial gyroscope and a tri-axial accelerometer. The UM6 (Clearpath Robotics, Kitchener, ON, Canada) magnetometer provided a static heading with an accuracy of better than 2°, serving as the initial heading for the system. Similarly, the GNSS receiver provided the initial longitude, latitude, and altitude.

Data processing was performed on a laptop equipped with an Intel i7-6700 CPU, a GT960m graphics card, and 12 GB of RAM. The operating system was Ubuntu 16.04, running the ROS (robot operating system) kinetic distribution. This software environment supports both sophisticated data simulation and advanced graphical rendering.

Table 4. The performance parameters of ADIS16445.

Performance	Parameters
Gyroscope Dynamic Range	$\pm 250^\circ/s$
Gyroscope Sensitivity	$0.01^\circ/s$
Gyroscope Nonlinearity	$\pm 0.1\%$
Gyroscope Bias Stability	$12^\circ/h$
Angular Random Walk	$0.56^\circ/\sqrt{h}$
Accelerometer Dynamic Range	± 5 g
Accelerometer Sensitivity	0.25 mg
Accelerometer Nonlinearity	$\pm 0.2\%$
Accelerometer Bias Stability	0.075 mg
Velocity Random Walk	$0.0735\ m/s/\sqrt{h}$
Bandwidth	330 Hz
Output Rate	100 Hz

4.3. Experimental Area

All data in the paper were collected in April 2023 at Harbin Engineering University and its surrounding areas, with geographic coordinates approximately at 126.68° longitude and 45.77° latitude and an elevation of about 130 m. Based on the actual driving environment, the driving speed of the SUV in different experiments was controlled between 15 km/h and 30 km/h.

4.4. Result and Analysis

As shown in Figure 12, this work compared two currently popular LiDAR SLAM methods with the algorithm proposed in the text. When the information is relatively rich, both LOAM and the algorithm presented in this paper achieved satisfactory results. However, due to the incorrect loop closure judgment at the end, SC-Lego-LOAM resulted in a certain deviation in the overall outcome. When the scene information was not sufficiently rich, such as in Figure 13, the LOAM algorithm exhibited attitude deviations at the end, which led to errors in the navigation results. In contrast, SC-Lego-LOAM encountered more severe errors in loop closure, rendering it entirely inoperative.

Data_1 was collected at 6 PM on 4 April 2023, near Building 61 of Harbin Engineering University, with a total traveled distance of 1460 m. In this scenario, the SUV's route was to circle around the building for two laps, and the main purpose of the scene setup was to verify the effectiveness of the algorithm in the paper under general environmental conditions. In the initial 1000 m, SC-Lego-LOAM maintained relatively good performance. However, after the final incorrect loop closure, the total distance was re-optimized, which led to a significant misalignment between the final distance and the azimuth angle. The method presented in this paper performed similarly to LOAM in the early stages, but because the original LOAM lacked loop closure detection functionality, its errors were bound to increase over time. Table 5 provides an overall summary of that experiment.

Data_2 was collected at 5 PM on 5 April 2023, near the parking lot of Harbin Engineering University, with a total traveled distance of 1403 m. As is shown in Figure 14 and Table 6, the structural feature weakened near the parking lot; the lack of structural features caused matching issues with the algorithm that relied on points. The experiment was mainly designed to demonstrate the stability of the algorithm in this paper relative to the comparative algorithms under the preset conditions of this paper. Judging from the comparison of results, the incorrect loop closure (red circle) by SC-Lego-LOAM led to severe issues once again, causing the method to fail entirely in this set of experiments. After the first loop closure, LOAM began to accumulate heading errors, which resulted in the continuous amplification of positioning errors in the subsequent SLAM due to the heading deviation.

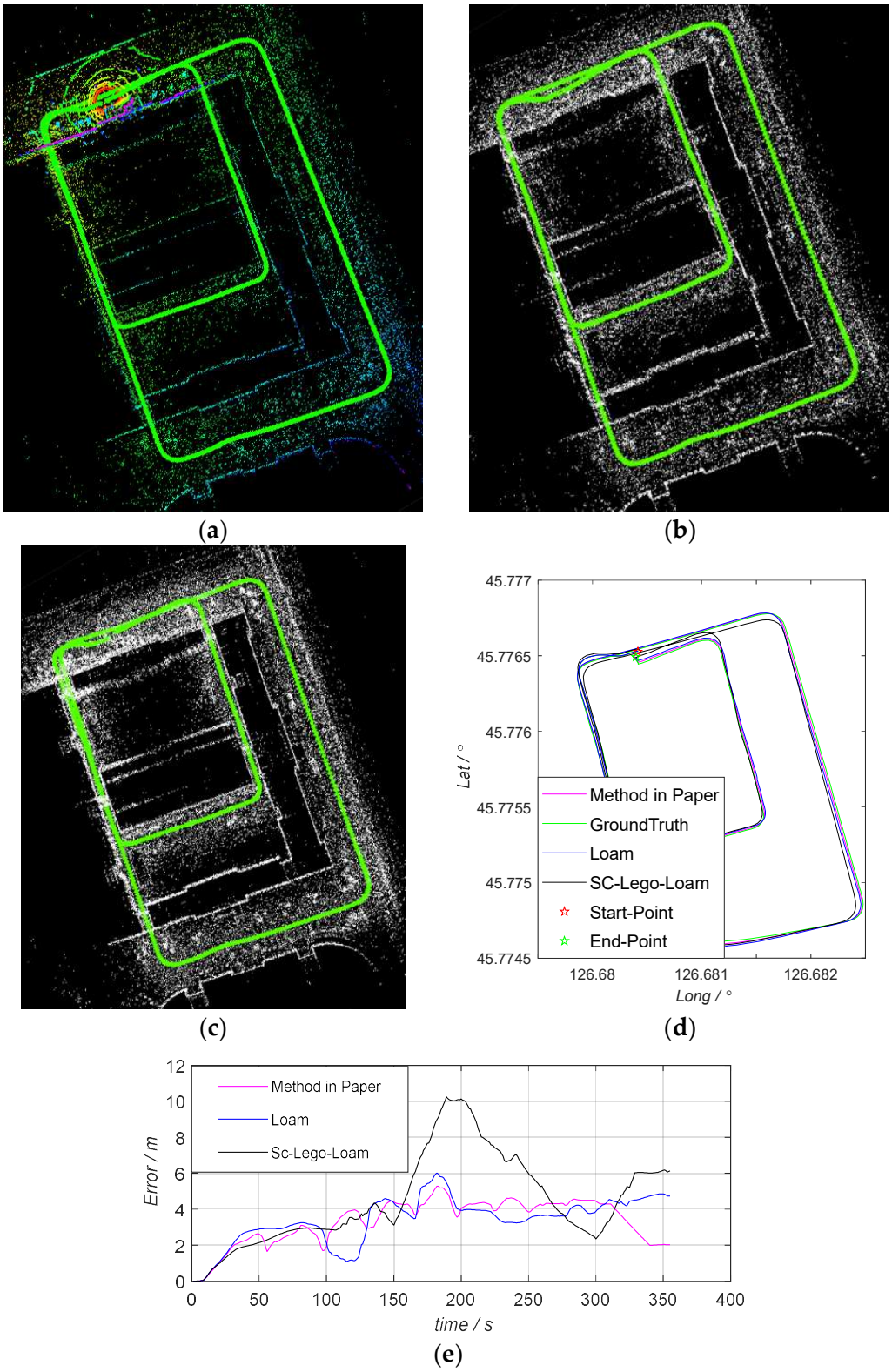


Figure 12. The result of LiDAR SLAM in Data_1: (**a**) was built by the LOAM; (**b**) was built by SC-Lego-LOAM; (**c**) shows the mapping result of the method from this paper; (**d**) depicts a direct comparison between various algorithms; (**e**) represents the positioning errors of the algorithms measured in meters.

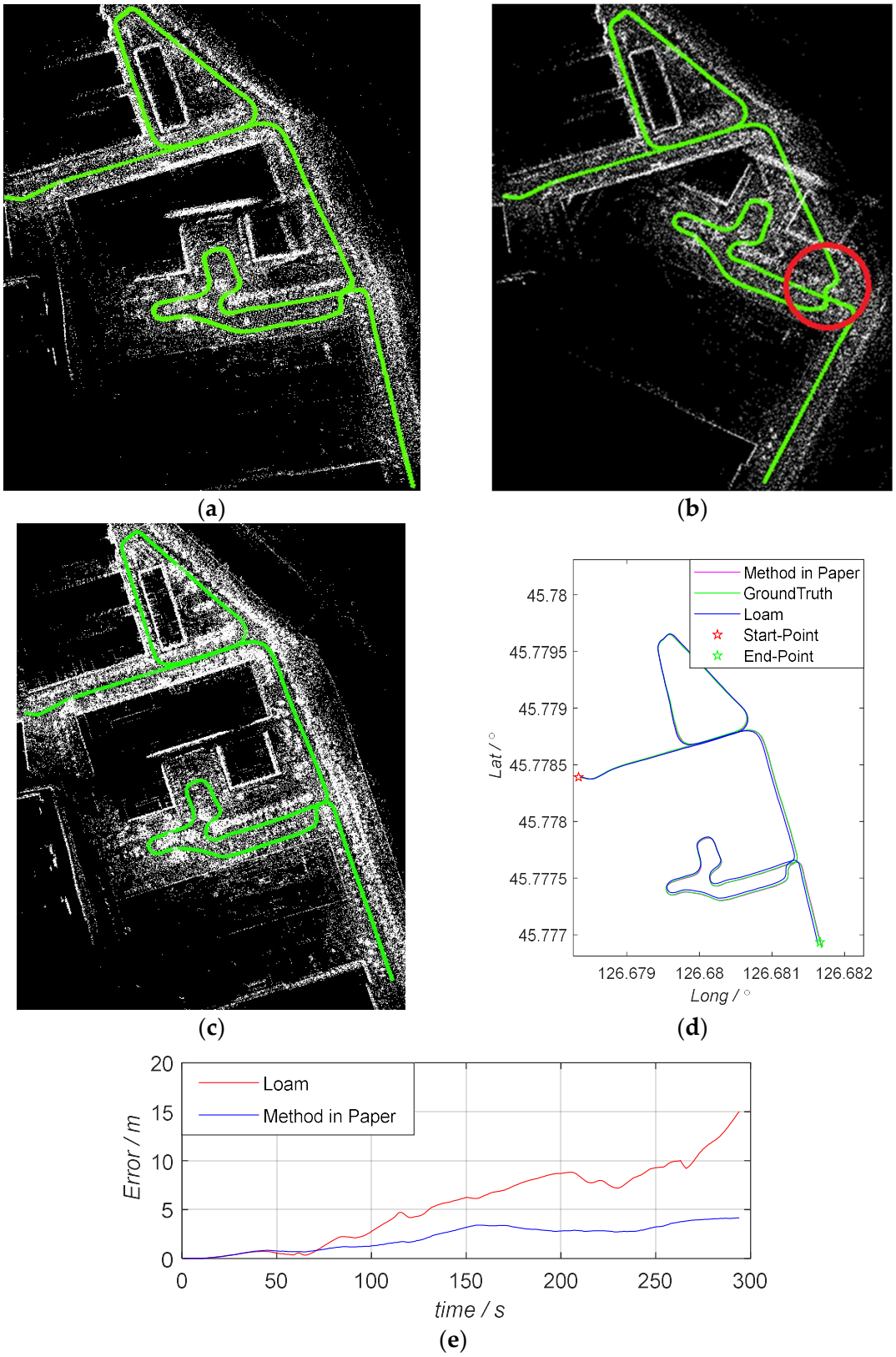


Figure 13. The result of LiDAR SLAM in Data_0405: (**a**) was built by the LOAM; (**b**) was built by SC-Lego-LOAM; (**c**) shows the mapping result of the method from this paper; (**d**) depicts a direct comparison between various algorithms; (**e**) represents the positioning errors of the algorithms measured in meters.

Table 5. The performance of SLAM methods for Data_1.

Method	LOAM	SC-Lego-LOAM	Method in Paper
Avg. Positioning Error (m)	3.89	4.14	3.33
Final Heading Error (°)	2.17	5.43	2.32
Max Positioning Error (m)	6.17	10.12	5.32
Travel Distance Error (m)	17	18	13

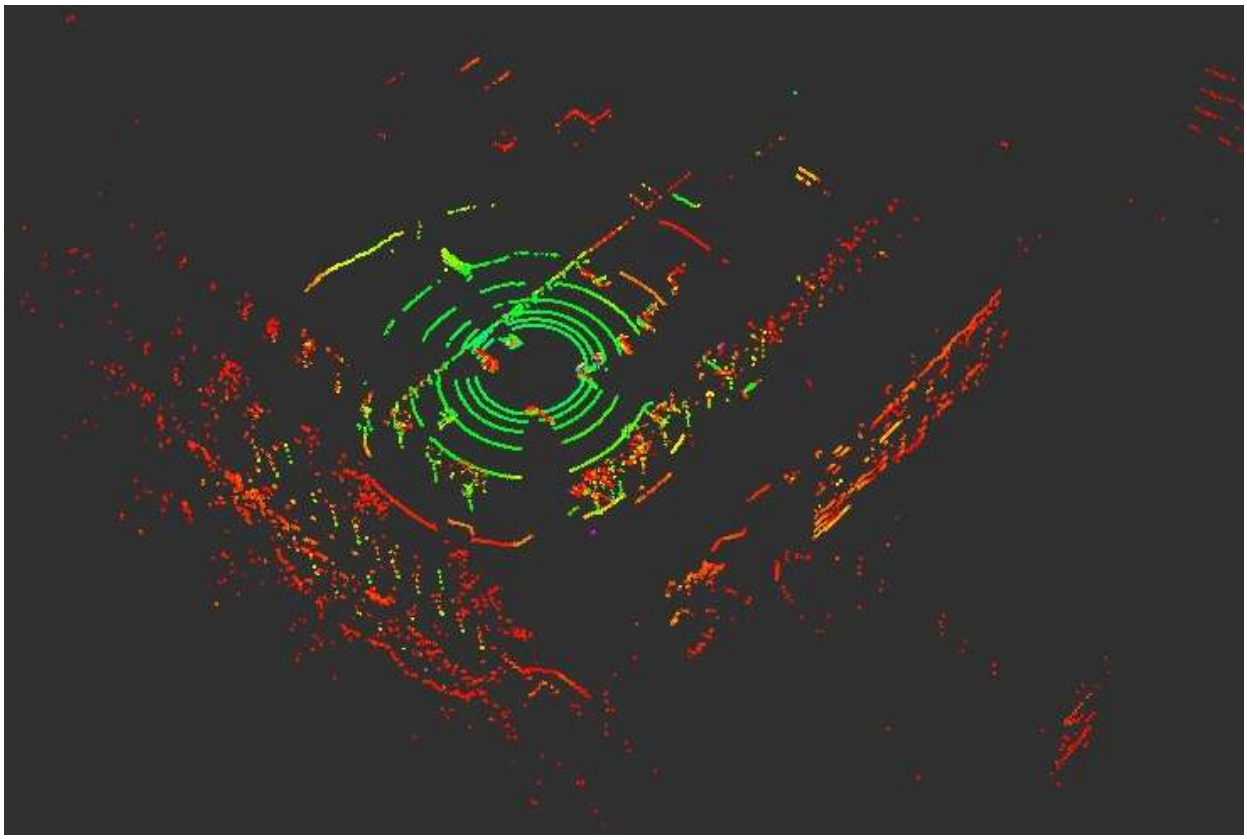


Figure 14. The point clouds near the parking lot.

Table 6. The performance of SLAM methods for Data_2.

Method	LOAM	SC-Lego-LOAM	Method in Paper
Avg. Positioning Error (m)	5.79	Failed	2.25
Final Heading Error (°)	6.22	Failed	1.02
Max Positioning Error (m)	15.17	Failed	4.12
Travel Distance Error (m)	22	Failed	11

The remaining data were also collected from 3–5 April 2023, at Harbin Engineering University and its surrounding areas. Data_3 had a loop closure point available shortly after the start, which could be used to eliminate accumulated errors. Data_4 featured a longer segment of nearly straight-line travel. These two sets of experiments were not designed intentionally with specific scenarios. They were standard test experiments; hence, they are not elaborately compared in detail within the text but are listed as supplementary experimental data in the Table 7.

Table 7. Comparison of positioning errors for other data.

Data	LOAM	SC-Lego-LOAM	Method in Paper	Travel Distance
Data_3	28.79 m	18.12 m	14.17 m	2205 m
Data_4	16.73 m	9.25 m	10.34 m	2234 m

5. Discussion

5.1. Results' Interpretation and Contribution

It redefines the fundamental computational unit in LiDAR SLAM by shifting the focus from LiDAR regression to an INS, rather than treating it as merely an accessory to LiDAR. The high-frequency output from the SINS navigation significantly reduced the computational load on the odometry component of LiDAR SLAM, thereby enhancing the accuracy of its positioning results.

In conventional scenarios, the performance of the algorithm proposed herein was comparable to that of LOAM. However, in scenarios where structural features were sparse or lacking, the algorithm demonstrated superior performance. The experimental results indicate that, while the algorithm achieved results similar to LOAM under typical conditions, it excelled in environments with limited structural features. In experiments that satisfied loop closure conditions, its relative advantage was even more pronounced. The relative accuracy improved by approximately 17%. From Figure 12, it can be observed that, in a general scenario, although the algorithm in the paper achieved good results, its basic performance was consistent with the other two algorithms. This scenario was only to verify the universal applicability of the algorithm in the paper, so there was no significant improvement in the specific comparative data. Figure 13 (DATA_2) is a preset scenario for the paper. Excluding the SC-Lego-LOAM algorithm, which was eliminated due to loop closure failure, from the error in Figure 13e, it can be seen that, after entering the parking lot environment, the error of the LOAM algorithm began to gradually increase, while the algorithm in the paper maintained a stable trajectory tracking. This fully demonstrated that the algorithm proposed in the paper achieved optimization for special scenarios while maintaining universality.

At the same time, this paper reorganizes and extracts the inherent characteristics of the point cloud. It goes a step further in the use of points, focusing the application of LiDAR point clouds on the edges that are less susceptible to the sparsity of point clouds and frequent changes in attitude matrices. This virtual edge composed of edge points is inherently a form of clustering. As long as points that meet the clustering criteria can be scanned, they can be continuously tracked in LiDAR SLAM and used to correct the positioning results of SINS. Of course, considering that the density of the point cloud has an attenuation characteristic with distance, in actual selection, further screening will only be carried out when a cluster contains at least four consecutive points and the line length exceeds 1 m. The selection of line distance rather than points, lines, or surfaces as the observation variable effectively reduces the computational load of the filter and increases the feasibility of real-time computation on low-performance devices.

For LiDAR SLAM loop closure based on scan context, the paper also makes certain rules changes. Experiments have proven that it can effectively reduce the occurrence of incorrect loop closure points and thereby enhance the overall accuracy of SLAM.

Although LiDAR SLAM algorithms based on machine learning have achieved excellent results, for in-vehicle processors, due to limitations in size and power, it is still difficult for their core to meet the real-time requirements in navigation. This work provides another feasible path within traditional algorithms.

5.2. Further Research

The experimental results presented in this paper demonstrate that the algorithm proposed within the text has superiority over traditional algorithms in both the odometry and mapping components of LiDAR SLAM. However, the results for Data_5 also indicate that in

complex long-distance environments, relying solely on LiDAR and SINS for navigation still cannot achieve long-term precise positioning. This suggests that the algorithm presented in the paper should only be used as a supplementary method to maintain the original navigation accuracy when GNSS signals are lost, rather than a complete substitute, in urban environments. In future research, exploring how to integrate GNSS-related data to further enhance the performance of LiDAR SLAM will be investigated.

Additionally, another point that requires attention is that, similar to other LiDAR algorithms, the divergence issue in the height channel of the algorithm presented in the paper has not been significantly improved. When the LiDAR is scanning in open areas (such as forest trails), it becomes extremely difficult to obtain lateral edge lines, and at this point, 3D LiDAR SLAM can degrade to a performance like that of 2D LiDAR SLAM. Furthermore, how to further subdivide and utilize ground points will also be one of the key research projects' focuses in the future.

6. Conclusions

This paper presents a novel LiDAR/SINS tightly integrated SLAM algorithm designed to address the localization challenges in urban environments characterized by sparse structural features. Building upon the LOAM framework, the algorithm introduces further processing of LiDAR point cloud classification to extract edge lines through clustering. Leveraging the rotational invariance of distance, the algorithm constructs a Kalman filter system based on the distance variation in edge lines. This approach contributes to enhanced robustness and positioning accuracy.

Experimental results obtained in local urban scenarios demonstrated a 17% enhancement in positioning accuracy when compared to traditional point-based methods, particularly in environments characterized by sparse features. By proposing a line distance-based observation model and detailing the associated EKF framework and parameter settings, the proposed method redefines the concepts of loosely and tightly coupled integration within LiDAR/SINS systems.

Future research will explore the integration of GNSS data to further enhance the performance of the proposed LiDAR SLAM system, particularly in complex and long-distance navigation scenarios. Additionally, key areas of focus for future work include improving performance in open areas, particularly in the vertical channel, and optimizing ground point utilization.

This study not only achieves significant algorithmic improvements over existing methods but also paves a new technological pathway for autonomous driving and robotic navigation applications.

Author Contributions: Conceptualization, X.X. and L.G.; methodology, X.X.; software, X.X.; validation, X.X., Y.C. and Z.L.; formal analysis, X.X. and L.G; investigation, X.X.; resources, X.X. and Z.L.; data curation, X.X.; writing—original draft preparation, X.X.; writing—review and editing, X.X. and L.G.; visualization, X.X. and Y.C.; supervision, L.G. and Y.G.; project administration, Y.G.; funding acquisition, L.G. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the Department of Science and Technology of Heilongjiang Province (2023ZX01A21) and the National Natural Science Foundation of China (NSFC. 61803118).

Data Availability Statement: The datasets presented in this article are not readily available because they are part of an ongoing study. Requests to access the datasets should be directed to the corresponding author.

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

Appendix A

The parameter settings for F in Equation (6) are as follows:

$$
\begin{aligned}
F_{11} &= \begin{bmatrix} 0 & 0 & -\frac{\dot{\phi}}{R_M+h} \\ \dot{\lambda}tan\phi & 0 & -\frac{\dot{\lambda}}{R_M+h} \\ 0 & 0 & 0 \end{bmatrix}, F_{23} = \begin{bmatrix} 0 & f_u & -f_n \\ -f_u & 0 & f_e \\ f_n & -f_e & 0 \end{bmatrix}, \\
F_{12} &= \begin{bmatrix} 0 & \frac{1}{R_M+h} & 0 \\ \frac{1}{(R_N+h)cos\phi} & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, F_{32} = \begin{bmatrix} 0 & \frac{1}{R_M+h} & 0 \\ \frac{-1}{R_N+h} & 0 & 0 \\ \frac{-tan\phi}{R_N+h} & 0 & 0 \end{bmatrix}, \\
F_{21} &= \begin{bmatrix} 2\omega_e(v_u sin\phi + v_n cos\phi) + \dot{\lambda}v_n/cos\phi & 0 & 0 \\ -2\omega_e v_e cos\phi - \dot{\lambda}v_e/cos\phi & 0 & 0 \\ -2\omega_e v_e sin\phi & 0 & 2g/R_N \end{bmatrix}, \\
F_{22} &= \begin{bmatrix} (v_n tan\phi - v_u)/(R_N+h) & \left(2\omega_e + \dot{\lambda}\right)sin\phi & -\left(2\omega_e + \dot{\lambda}\right)cos\phi \\ -2\omega_e v_e cos\phi - \dot{\lambda}v_e/cos\phi & -v_u/(R_M+h) & -\dot{\phi} \\ -2\omega_e v_e sin\phi & 2\dot{\phi} & 0 \end{bmatrix}, \\
F_{31} &= \begin{bmatrix} 0 & 0 & -\dot{\phi}/(R_M+h) \\ \omega_e sin\phi & 0 & \dot{\lambda}cos\phi/(R_N+h) \\ -\omega_e cos\phi - \dot{\lambda}/(R_N+h)cos\phi & 0 & \dot{\lambda}sin\phi/(R_N+h) \end{bmatrix}, \\
F_{44} &= \begin{bmatrix} -\beta_{\omega x} & 0 & 0 \\ 0 & -\beta_{\omega y} & 0 \\ 0 & 0 & -\beta_{\omega z} \end{bmatrix}, F_{55} = \begin{bmatrix} -\beta_{fx} & 0 & 0 \\ 0 & -\beta_{fy} & 0 \\ 0 & 0 & -\beta_{fz} \end{bmatrix}, \\
F_{33} &= \begin{bmatrix} 0 & \left(\omega_e + \dot{\lambda}\right)sin\phi & -\left(\omega_e + \dot{\lambda}\right)cos\phi \\ -\left(\omega_e + \dot{\lambda}\right)sin\phi & 0 & -\dot{\phi} \\ \left(\omega_e + \dot{\lambda}\right)cos\phi & \dot{\phi} & 0 \end{bmatrix}
\end{aligned} \tag{A1}
$$

Table A1. The positions of different sensors relative to the FOG center.

Sensors	Right	Forward	Up
MEMS	5.3 cm	−22.5 cm	−4.3 cm
LiDAR	−32.5 cm	78 cm	113.5 cm
GNSS Antenna	21.5 cm	68.5 cm	83 cm

References

1. Montemerlo, M.; Thrun, S.; Koller, D.; Wegbreit, B. Fast SLAM: A factored solution to the simultaneous localization and mapping problem. In Proceedings of the AAAI-02: Eighteenth National Conference on Artificial Intelligence, Edmonton, AL, Canada, 28 July–1 August 2002; Volume 593598.
2. Huang, L. Review on LiDAR-based SLAM techniques. In Proceedings of the 2021 International Conference on Signal Processing and Machine Learning (CONF-SPML), Stanford, CA, USA, 14 November 2021; pp. 163–168.
3. Cadena, C.; Carlone, L.; Carrillo, H.; Latif, Y.; Scaramuzza, D.; Neira, J.; Reid, I.; Leonard, J.J. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Trans. Robot.* **2016**, *32*, 1309–1332. [CrossRef]
4. Biber, P.; Straßer, W. The normal distributions transform: A new approach to laser scan matching. In Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No. 03CH37453), Las Vegas, NV, USA, 27–31 October 2003; Volume 3, pp. 2743–2748.
5. Rusinkiewicz, S.; Levoy, M. Efficient Variants of the ICP Algorithm. In Proceedings of the of the Third International Conference on 3-D Digital Imaging and Modeling, Quebec City, QC, Canada, 28 May–1 June 2001; pp. 145–152.
6. Hung, Y.-W.; Chen, Y.-C.; Lo, C.; So, A.G.; Chang, S.-C. Dynamic workload allocation for edge computing. *IEEE Trans. Very Large-Scale Integr. (VLSI) Syst.* **2021**, *29*, 519–529. [CrossRef]
7. Deng, Q.; Sun, H.; Chen, F.; Shu, Y.; Wang, H.; Ha, Y. An Optimized FPGA-Based Real-Time NDT for 3D-LiDAR Localization in Smart Vehicles. *IEEE Trans. Circuits Syst. II: Express Briefs* **2021**, *68*, 3167–3171. [CrossRef]
8. Jiang, M.; Song, S.; Li, Y.; Liu, J.; Feng, X. Scan registration for mechanical scanning imaging sonar using kD2D-NDT. In Proceedings of the 2018 Chinese Control and Decision Conference (CCDC), Shenyang, China, 9–11 June 2018; pp. 6425–6430.

9. Zhang, J.; Singh, S. LOAM: Lidar odometry and mapping in real-time. *Robot. Sci. Syst.* **2014**, *2*, 1–9.
10. 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. [CrossRef]
11. He, L.; Wang, X.; Zhang, H. M2DP: A novel 3D point cloud descriptor and its application in loop closure detection. In Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, Republic of Korea, 9–14 October 2016. [CrossRef]
12. Kim, G.; Kim, A. Scan Context: Egocentric Spatial Descriptor for Place Recognition within 3D Point Cloud Map. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 4802–4809. [CrossRef]
13. Kim, G.; Choi, S.; Kim, A. Scan Context++: Structural Place Recognition Robust to Rotation and Lateral Variations in Urban Environments. *IEEE Trans. Robot.* **2022**, *38*, 1856–1874. [CrossRef]
14. 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 on Intelligent Robots and Systems (IROS), Prague, Czech Republic, 27 September–1 October 2021; pp. 4390–4396.
15. Tang, J.; Chen, Y.; Niu, X.; Wang, L.; Chen, L.; Liu, J.; Shi, C.; Hyyppa, J. Lidar scan matching aided inertial navigation system in gnss-denied environments. *Sensors* **2015**, *15*, 16710–16728. [CrossRef] [PubMed]
16. Xu, X.; Zhang, L.; Yang, J.; Cao, C.; Wang, W.; Ran, Y.; Tan, Z.; Luo, M. A Review of Multi-Sensor Fusion SLAM Systems Based on 3D LIDAR. *Remote Sens.* **2022**, *14*, 2835. [CrossRef]
17. Koide, K.; Yokozuka, M.; Oishi, S.; Banno, A. Globally consistent and tightly coupled 3D LiDAR inertial mapping. In Proceedings of the 2022 International Conference on Robotics and Automation (ICRA), Philadelphia, PA, USA, 23–27 May 2022; pp. 5622–5628.
18. 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.
19. Rabbou, M.A.; El-Rabbany, A. Tightly coupled integration of GPS precise point positioning and MEMS-based inertial systems. *GPS Solut.* **2015**, *19*, 601–609. [CrossRef]
20. Rong, H.; Gao, Y.; Guan, L.; Ramirez-Serrano, A.; Xu, X.; Zhu, Y. Point-Line Visual Stereo SLAM Using EDlines and PL-BoW. *Remote Sens.* **2021**, *13*, 3591. [CrossRef]
21. Wang, H.; Guan, L.; Yu, X.; Zhang, Z. PL-ISLAM: An Accurate Monocular Visual-Inertial SLAM with Point and Line Features. In Proceedings of the 2022 IEEE International Conference on Mechatronics and Automation (ICMA), Guilin, China, 7–10 August 2022; pp. 1141–1146. [CrossRef]
22. Mourikis, A.I.; Roumeliotis, S.I. A multi-state constraint Kalman filter for vision-aided inertial navigation. In Proceedings of the 2007 IEEE International Conference on Robotics and Automation, Roma, Italy, 10–14 April 2007; pp. 3565–3572.
23. 32/16-Line Mechanical Line Mechanical LiDAR | Leishen Intelligent System. Available online: https://www.lslidar.com/product/c32-16-mechanical-lidar/ (accessed on 6 July 2024).
24. Himmelsbach, M.; Hundelshausen, F.V.; Wuensche, H.-J. Fast Segmentation of 3D Point Clouds for Ground Vehicles. In Proceedings of the IEEE Intelligent Vehicles Symposium, La Jolla, CA, USA, 21–24 June 2010; pp. 560–565.
25. Fischler, M.A.; Bolles, R.C. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM* **1981**, *24*, 381–395. [CrossRef]
26. Bogoslavskyi, I.; Stachniss, C. Fast Range Image-based Segmentation of Sparse 3D Laser Scans for Online Operation. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Daejeon, Republic of Korea, 9–14 October 2016; pp. 163–169.
27. Arthur, D.; Vassilvitskii, S. k-means++: The advantages of careful seeding. In Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete algorithms (SODA), New Orleans, LA, USA, 7–9 January 2007; Volume 7, pp. 1027–1035.
28. Belongie, S.; Malik, J.; Puzicha, J. Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Anal. Mach. Intell.* **2002**, *24*, 509–522. [CrossRef]
29. Guan, L.; Cong, X.; Sun, Y.; Gao, Y.; Iqbal, U.; Noureldin, A. Enhanced MEMS SINS aided pipeline surveying system by pipeline junction detection in small diameter pipeline. *IFAC-Pap.* **2017**, *50*, 3560–3565. [CrossRef]