remote sensing

MDPI

Article

Planar Constraint Assisted LiDAR SLAM Algorithm Based on Manhattan World Assumption

Haiyang Wu [1,2,3,†], Wei Wu [4,†], Xingyu Qi [1,2,3], Chaohong Wu [1,2,3], Lina An [1,2,3] and Ruofei Zhong [1,2,3,*]

1 Key Laboratory of 3D Information Acquisition and Application, MOE, Capital Normal University, Beijing 100048, China
2 Base of the State Key Laboratory of Urban Environmental Process and Digital Modeling, Capital Normal University, Beijing 100048, China
3 College of Resource Environment and Tourism, Capital Normal University, Beijing 100048, China
4 School of Vehicle and Mobility, Tsinghua University, Beijing 100084, China
* Correspondence: zrf@cnu.edu.cn
† These authors contributed equally to this study and shared the first authorship.

Abstract: Simultaneous localization and mapping (SLAM) technology based on light detection and ranging (LiDAR) sensors has been widely used in various environmental sensing tasks indoors and outdoors. However, it still lacks effective constraints in structured environments such as corridors and parking lots, and its accuracy needs improvement. Based on this, a planar constraint-assisted LiDAR SLAM algorithm based on the Manhattan World (MW) assumption is proposed in this paper. The algorithm extracts planes from the environment point cloud submap, classifies the planes according to the ground and vertical planes, and calculates the main direction angles of the ground and vertical plane, respectively, to construct constraints. To enhance the stability and robustness of the system, a two-step main direction angle calculation and update strategy are designed, and a hysteresis update is used to avoid the introduction of errors by unoptimized planes. This paper uses a backpack laser scanning system to collect experimental data in various scenes. These data are used to compare our method with three open-source LiDAR SLAM algorithms, that are currently more widely used and perform better. Qualitative and quantitative experiments are conducted to verify the effectiveness of our method. The experimental results show that the absolute accuracy of the point clouds obtained by our method is improved by 77.46% on average compared with the other three algorithms in the environment, conforming to the MW assumption, which verifies the effectiveness of the algorithm.

Keywords: simultaneous localization and mapping (SLAM); planar constraint; Manhattan World assumption; indoor scenes

Citation: Wu, H.; Wu, W.; Qi, X.; Wu, C.; An, L.; Zhong, R. Planar Constraint Assisted LiDAR SLAM Algorithm Based on Manhattan World Assumption. *Remote Sens.* **2023**, *15*, 15. https://doi.org/10.3390/rs15010015

Academic Editor: Henning Buddenbaum

Received: 16 November 2022
Revised: 16 December 2022
Accepted: 17 December 2022
Published: 21 December 2022

1. Introduction

Three-dimensional spatial information is an essential part of geographic information. With the development of light detection and ranging (LiDAR) sensors, it has been possible to acquire high-precision 3D LiDAR point cloud data in outdoor environments with good global navigation satellite system (GNSS) signals based on LiDAR combined with GNSS and the inertial navigation system (INS) or inertial measurement unit (IMU) technologies. However, acquiring high-precision 3D LiDAR point cloud data in indoor and other GNSS-denied environments is still one of the research hotspots. In recent years, the development of simultaneous localization and mapping (SLAM) technology has brought new ideas to solve this hotspot problem. SLAM refers to simultaneous localization and mapping, a technique for the carrier pose estimation and map construction in unknown environments. According to the different sensors used, SLAM can be divided into visual SLAM based on sensors such as cameras and LiDAR SLAM based on LiDAR sensors. Visual SLAM [1,2] has high localization accuracy but is more sensitive to changes in illumination and the viewing angle. LiDAR SLAM is not affected by changes in illumination and viewing angle,

and LiDAR has a high-ranging accuracy. LiDAR can be divided into mechanical LiDAR and non-mechanical LiDAR according to the internal structure, and mechanical LiDAR can be divided into single-line LiDAR and multi-line LiDAR. Compared with 2D LiDAR SLAM using single-line LiDAR, 3D LiDAR SLAM using multi-line LiDAR can provide richer 3D environmental information.

In the field of 3D LiDAR SLAM, scholars have proposed many excellent algorithms. One of the most influential algorithms is the LiDAR odometry and mapping (LOAM) algorithm [3], which views 3D LiDAR SLAM as a framework combining a high-frequency, low-accuracy front-end module with a low-frequency, high-precision back-end module. In the front-end module, feature points (both edge point and surface point types) are extracted for each frame of acquired point cloud data, and the extracted feature points are later used to match between frames to construct the LiDAR odometry. In the back-end module, the bit pose of the front-end odometer output is optimized, and the environmental map is constructed using the results from multiple scan matching. The LOAM algorithm does not have a loop closure module and cannot effectively constrain the cumulative error. The LIO-SAM [4] algorithm further improves the LOAM algorithm by constructing a tightly coupled LiDAR-IMU odometry framework with factor maps and supporting the introduction of GNSS observations to correct cumulative errors and using IMU observations to motion compensate the LiDAR observations to remove distortions. The algorithm achieves real-time performance using a sliding window-based scan-matching method. The LIO-SAM algorithm has obtained higher accuracy point cloud maps in some outdoor scenarios with the help of GNSS to constrain the localization error. However, in indoor scenarios without a priori constraints or scenarios with poor GNSS signals, only loop closure constraints can be relied on to correct the accumulated positional estimation errors. A Euclidean distance threshold triggers the loop closure constraint, which needs to be more robust. The authors of [5] introduced descriptors to improve the loop closure part of LIO-SAM to enhance the robustness and speed of the loop closure part of the original algorithm, but the loop closure constraint will only be triggered when revisit occurs, and the constraint is limited for large scenes or no revisit environments. Unlike the LOAM algorithm and its variants, FAST-LIO2 [6] does not extract feature points for each acquired LiDAR point cloud frame and directly align the original point cloud with the point cloud submap. It introduces more point cloud features to improve the matching accuracy and uses an incremental k-d tree structure to ensure computational efficiency to achieve real-time performance. However, FAST-LIO2 does not have a loop closure module and does not support the introduction of GNSS observations, so it lacks effective constraints in large open scenarios.

Although the above methods have high accuracy in environments with good GNSS signals or rich features, high-accuracy point cloud maps cannot be obtained in structured and GNSS-denied environments such as corridors and parking lots. Many studies have attempted to address this issue. The LeGO-LOAM [7] algorithm uses a segmentation clustering method to extract the ground in the front-end part for the ground unmanned vehicle LiDAR parallel to the ground, increasing the constraint on the ground and reducing the error in the z-direction. However, the algorithm requires that the LiDAR be parallel to the ground and only apply to some ground unmanned vehicles. Research by [8–10] added geometric feature constraints such as lines and planes to improve the robustness and accuracy of the system in vision-based and LiDAR-based SLAM frameworks, respectively. The research by [11] also focuses on geometric feature constraints. It used the nearest point representation proposed in [10] to represent the plane, successively fitted the plane corresponding to the ground point cloud and constructed the ground constraint based on the plane relationship of the fitted ground between two different frames to improve the elevation offset problem of the LiDAR SLAM estimated poses in the indoor parking environment.

The above method constructs constraints by introducing geometric information. Although it has some positive effects, the improvement in accuracy is still limited because the geometric features used to construct the constraints come from the current local submap.

Cumulative errors have been introduced during the construction of the local submap. A structured environmental assumption can be introduced to better constrain the pose estimation of SLAM using geometric information in the current environment, such as the MW assumption [12]. The introduction of the MW assumption is a presupposition of the environment, so it avoids the introduction of geometric constraints along with the introduction of pre-existing cumulative errors. The MW assumption proposes the existence of structured features in the artificial environment, i.e., the existence of a large number of parallel and perpendicular relationships, which can be summarized as the existence of three main directions in the environment where the artificial buildings are always identical and orthogonal to each other. In the real world, many structured scenes (e.g., parking lots, corridors, etc.) can be regarded as conforming to the MW assumption, so the three main directions of the MW assumption can be used to assist and improve SLAM in these scenes. In the field of visual-based SLAM, many researchers have already conducted in-depth research on visual SLAM algorithms aided by the MW assumption. The research by [13–16] improved the visual-based SLAM framework based on the structured features of the environment expressed by the MW assumption to improve the drift and increase the accuracy in low-texture environments. The authors of [17] used structured lines as new features in the MW to complete map construction and localization, and the accuracy and robustness of the system were improved due to the introduction of directionality constraints. The research by [18] proposed a visual-inertial odometer based on multiple Manhattan Worlds overlaying the environment, further improving the generality of the method. In the field of LiDAR SLAM, there are still few related studies. The authors of [19] used the MW assumption to extract orthogonal planes and generate plane-based maps, which are more convenient for subsequent path planning while occupying less memory. However, the algorithm is more focused on the real-time performance of the system rather than accuracy.

In the MW assumption-based SLAM approach, one core task is extracting the planes in the current environment to construct screen constraints. The random sample consensus (RANSAC) algorithm [20,21] is a commonly used method for plane extraction, which is more robust to outliers and still gives good extraction results in the presence of a small number of outliers. For the case where the environment contains multiple planes, the point cloud can also be segmented using the Hough transform [22] or a point cloud segmentation method such as the region growth method [23] before plane extraction.

Considering the above problems and existing studies, this paper introduces the MW assumption to express the regular geometric features in the environment. We propose a planar constraint method based on LiDAR in the environment conforming to the MW assumption. This method adopts different processing strategies for the ground and vertical planes, such as walls and columns. It divides the six-degree freedom pose estimation problem of {roll, pitch, yaw, x, y, z} into two sub-problems {roll, pitch, z} three-degree of freedom pose estimation and {yaw, x, y} three-degree of freedom pose estimation, which improves the accuracy and robustness of the system. Furthermore, in the non-MW assumption environment, the present method will stop functioning due to the non-detection of a plane that matches the threshold. Still, it will not affect the regular operation of the rest of the SLAM system. The main contributions of this paper are as follows:

1. A planar constraint method based on the MW assumption for ground and vertical planes separately is designed to introduce planar constraints in the LiDAR SLAM framework.
2. A two-step strategy for calculating and updating the main direction angles for the vertical plane is used to avoid the introduction of errors in the unoptimized plane by lagging the update of the main direction angles.
3. Based on the backpack laser scanning system, we collected real data in the indoor parking lot, corridor, and outdoor environments and used the mapping method to obtain the absolute coordinates of the manually set target points in the above environments. The quality and absolute accuracy of the point clouds obtained by three LiDAR SLAM algorithms with good performance and wide application at present, and our method in the above dataset are evaluated. The experimental results

show that this method achieves better accuracy in all environments conforming to the MW assumption.

The remainder of this paper is organized as follows: Section 2 proposes a method for constructing planar constraints based on the MW assumption. Section 3 verifies the proposed method using a backpack laser scanning system to collect data. Section 4 provides a discussion of the proposed method in this paper. Finally, Section 5 concludes this study.

2. Methodology

The overall flow of the planar constraint algorithm based on the MW assumption is shown in Figure 1. The method in this paper is implemented based on the LIO-SAM [4] algorithm framework. For the raw data containing LiDAR data, IMU data, and GNSS data, each frame of the point cloud is first processed by the front-end module of the LIO-SAM algorithm. Whenever k frames of point clouds are accumulated, they are combined into a point cloud submap. For this submap, three steps are performed: (1) calculate the main direction angle; (2) construct the planar constraint; (3) solve the planar constraint, to add the planar constraint to the system. The planar constraint optimizes the results of the pose estimation together with other constraints in LIO-SAM such as closed-loop, and ultimately, constructs a globally consistent optimized point cloud map. The specific procedure of the method in this paper is explained in this section.

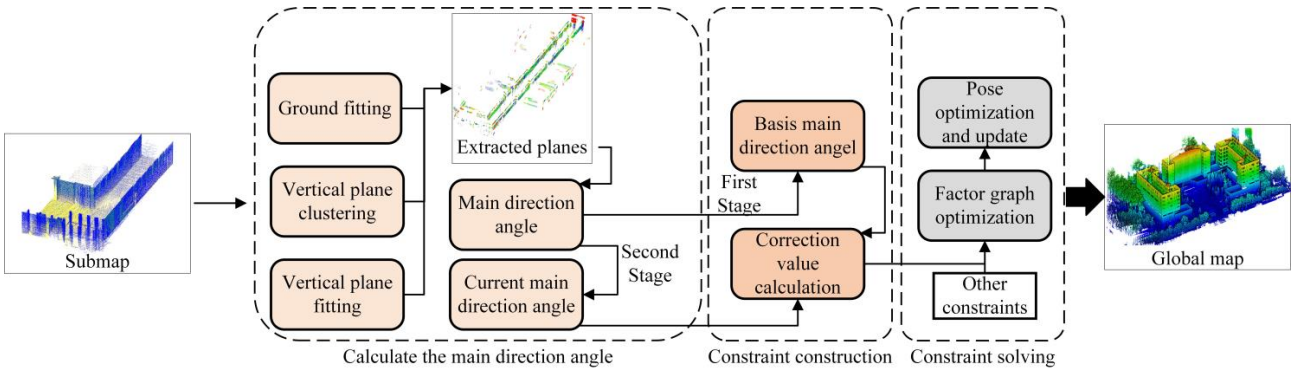


Figure 1. The overall flow of the planar constraint algorithm.

2.1. Main Direction Angle Calculation

This part corresponds to the main direction angle calculation module shown in the dashed box in Figure 1. In this part, the point cloud submap previously obtained by merging the k-frame point clouds will be processed, and the main direction angle of the current point cloud submap will be calculated. The main direction angle will be used to construct subsequent planar constraints.

2.1.1. Pre-Processing

LiDAR acquires many raw points in data acquisition, which brings redundant information and increases the computational effort in point cloud processing. This paper divides the three mutually orthogonal Manhattan main directions into one main longitudinal direction and two main horizontal directions. To obtain the two types of main directions, plane fitting and normal estimation are required for the vertical and ground planes, respectively. Removing the point clouds at the floor and ceiling when processing the vertical plane can improve the processing efficiency without affecting accuracy. Similarly, removing the rest of the point clouds when processing the ground will also improve efficiency. Therefore, the direct-pass filtering of the point cloud submap according to Equation (1), and the point

clouds within a specific range near (x_w, y_w, z_w) are reserved as the ground submap and the vertical plane submap respectively:

$$\begin{cases} \left|x_w - p_x^i\right| < Thr_x^{filter} \\ \left|y_w - p_y^i\right| < Thr_y^{filter} \\ \left|z_w - p_z^i\right| < Thr_z^{filter} \end{cases} \tag{1}$$

where (x_w, y_w, z_w) are the carrier position corresponding to the center frame of the input point cloud submap, p^i is the $i-$th point in the point cloud submap, $Thr^{filter} = \left\{Thr_x^{filter}, Thr_y^{filter}, Thr_z^{filter}\right\}$ is the threshold value, the threshold value can be set according to the type of the point cloud submap.

Using the pass-through filter for the point cloud can reduce the computation while ensuring the algorithm's accuracy. Still, it cannot effectively remove the possible outliers in the point cloud, so it is also necessary to perform statistical filtering of the cropped point cloud based on the Pauta criterion to remove the outliers. That is, the average distance from each point in the point cloud submap to other points is calculated, the average distance of each point in the point cloud conforms to the Gaussian distribution, and the mean value of the Gaussian distribution is calculated. If the mean distance of the point is more significant than three times the mean value, the point is considered an outlier and is removed.

2.1.2. Plane Fitting

Based on the MW assumption, the ground in different regions within the point cloud submap can be considered parallel. In the real world, the ground within a small area of a non-stair region is usually not only parallel but also coplanar. That is, within a point cloud submap, the ground equations can be considered unique. To make this assumption hold better, the number of frames that constitute the point cloud submap can be reduced to limit the range of the point cloud submap.

After pre-processing, the ground and vertical submaps are obtained for convenient computation. These submaps are stored in point clouds, which only facilitate the direct description of the planes while taking up many memory resources. Therefore, it is necessary to use a mathematical model, Equation (2), to fit the point cloud submaps, and the fitting is performed as described below:

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

where A, B, C, and D are the plane coefficients.

Firstly, the ground plane submap will be processed. The RANSAC is used in this paper to find the parameters of the fitting plane of the ground plane submap. The RANSAC algorithm is commonly used to estimate the mathematical model from a set of data that best fits the set of data [24], and the RANSAC algorithm can robustly estimate the mathematical model even if the data contains a portion of outlier points. The RANSAC algorithm works by continuously adjusting the parameters of the fitting plane to eliminate the outer points whose distances to the fitting plane are greater than a threshold and to retain the inner points whose distances are less than a threshold. At the end of the iteration, we use the final number of interior points and the mean absolute error (MAE) value to evaluate the accuracy of the fit [25], where MAE is defined as shown in Equation (3). Determine whether to use the results of this ground fit according to the rules in Equation (4):

$$MAE = \frac{\sum_{i=1}^{n} \frac{|A*x_i + B*y_i + C*z_i + D|}{\sqrt{A^2+B^2+C^2}}}{n} \tag{3}$$

$$\begin{cases} MAE < Thr_{MAE} \\ n > Thr_n \end{cases} \tag{4}$$

where (x_i, y_i, z_i) is the coordinate of the $i-$th point in the point cloud formed by the current interior points and n is the current number of interior points. Thr_{MAE} and Thr_n are the threshold values.

Next, the vertical plane submap will be processed. There are usually multiple planes within a vertical plane submap, and these planes are generally not co-planar but only have spatial relationships, such as being perpendicular or parallel. Therefore, when fitting the vertical plane submaps, it is necessary to partition the vertical plane submaps. This paper uses the region-growing method to segment the vertical plane submap. First, estimate the norm of each point in the vertical plane submap based on a principal component analysis (PCA) and rank the curvature of each point. Then, the point with the minor curvature is selected as the initial seed point of the region growth algorithm so that the segmentation can start from the smoothest region and can improve the efficiency of the region growth algorithm. After obtaining the segmentation results of the vertical plane, submaps use the RANSAC algorithm for each segmented vertical plane to find the parameters of the fitted plane and use the rules in Equation (4) to determine whether to use the results of this vertical plane fitting.

2.1.3. Calculate the Main Direction Angle

In the MW, the man-made buildings in the environment have three main directions that are orthogonal to each other, including one ground main direction and two vertical plane main directions that are perpendicular to the ground main direction and orthogonal to each other. In this paper, the two vertical plane main directions are converted into one main direction by their orthogonal relationship, which facilitates the construction of subsequent planar constraints.

As shown in Figure 2b, the plane normal vector can be obtained from the plane equation after obtaining the fitted equations of the two planes, the ground, and vertical plane. The normal vector of the plane (Equation (2)) is $\vec{n} = (A, B, C)$. In this paper, the spatial Cartesian coordinate system adopts the carrier coordinate system, i.e., the coordinate system of horizontal LIDAR$(\mathrm{X} - \mathrm{Y} - \mathrm{Z})$, as the global coordinate system, which is schematically shown in Figure 3.

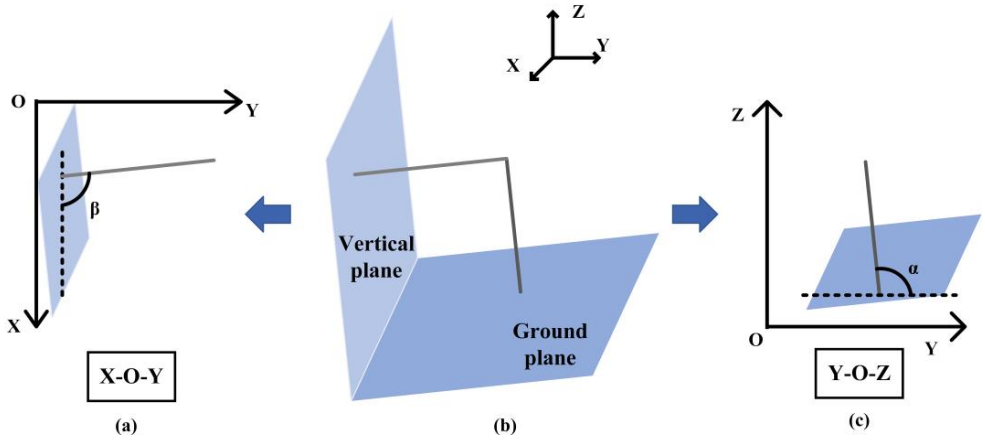


Figure 2. Schematic diagram of the main directional angle projection: (**a**) is the projection of the vertical plane normal in the XOY plane; (**b**) is the schematic diagram of the ground, vertical plane, and their normals; (**c**) is the projection of the ground normal in the YOZ plane.

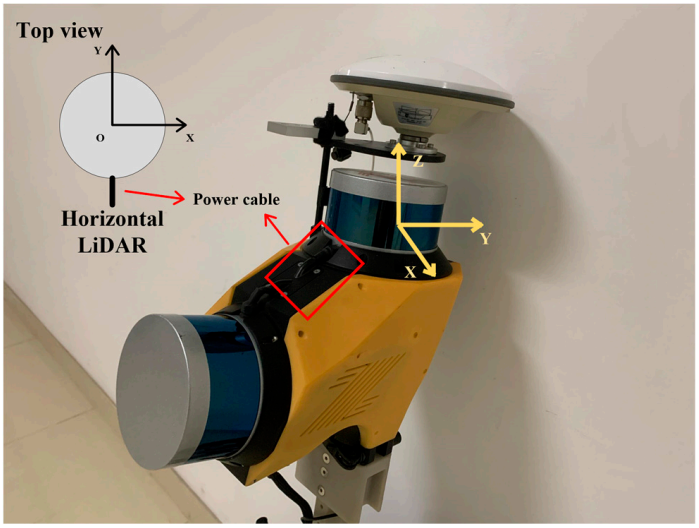


Figure 3. Schematic diagram of the global coordinate system. This paper uses the horizontal LiDAR coordinate system as the global coordinate system. The horizontal LiDAR coordinate system is a right-handed Cartesian coordinate system with the power line orthogonal to the Y direction, the vertical LiDAR plane upward to the Z direction, and the X direction orthogonal to the Y direction.

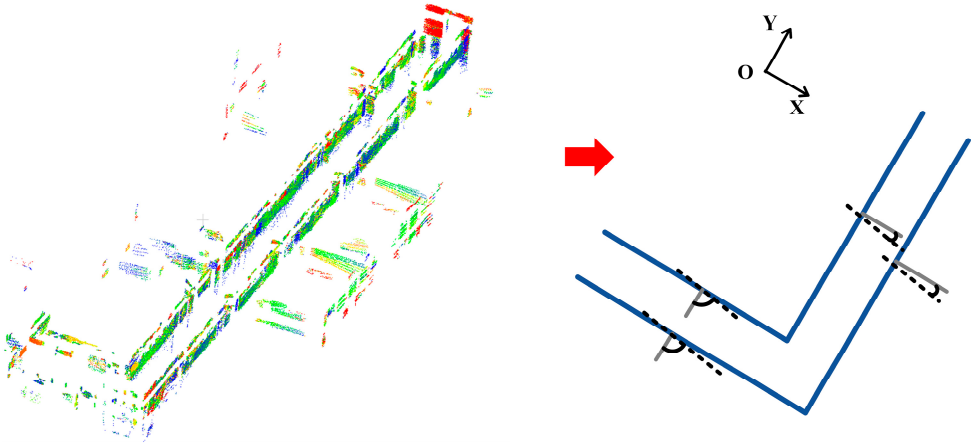
In calculating the main direction angle of the plane, we specify the main direction angle of the vertical plane as the angle between the normal vector of the vertical plane and the OX axis after projecting it to the XOY plane of the global coordinate system (angle β in Figure 2a). The main direction angle of the ground is also defined as the angle between the normal vector of the ground and the OY axis after projection to the YOZ plane of the global coordinate system (angle α in Figure 2c). The angle α and angle β are defined in Equation (5):

$$\begin{cases} \alpha = \arctan\frac{C}{B} \\ \beta = \arctan\frac{B}{A} \end{cases} \tag{5}$$

where A, B, and C are the coefficients of the plane (Equation (2)). The calculated angle α and angle β take values in the range of $\left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$.

Since the direction of the normal vector obtained when using the RANSAC algorithm to find the fitting parameters of the plane has a duality, the principal direction angle calculated in the previous step needs to be further processed. Firstly, the ground main direction angle will be processed. In this paper, only one ground submap is selected within the scope of a point cloud submap, so there is only one ground main direction angle corresponding to it. If the previously calculated ground main direction angle is negative, add π to the angle to transform it into its opposite direction. Convert the ground main direction angle to a positive value.

Next, the vertical plane main direction angles are processed. As shown in Figure 4, there may be multiple vertical planes in a point cloud submap, and according to the MW, the horizontal main direction angles of these planes should be the same. However, in the actual calculation, there may be mis-segmentation in the plane segmentation due to an error in fitting the plane. Some vertical planes in the scene do not conform to the MW assumption, so it is necessary to merge the horizontal main direction angles of multiple vertical planes in a submap and eliminate a small number of vertical planes that do not conform to the MW assumption.



(**a**) Extracted planes (**b**) Vertical plane main direction angle

Figure 4. Schematic diagram of the main direction angles of multiple vertical planes within one submap: (**a**) shows a schematic diagram of the extracted vertical planes in the corridor scenario (shown by stitching together the extracted planes from multiple vertical plane submaps); (**b**) shows a schematic diagram of the extracted planes corresponding to a point cloud submap.

The two mutually orthogonal vertical plane main direction angles can be converted to a unified vertical plane main direction angle α by the conversion rule in Equation (6), $\alpha \in \left[-\frac{\pi}{4}, \frac{\pi}{4}\right]$:

$$\alpha = \begin{cases} \alpha - \frac{\pi}{2}, & \left(\alpha > \frac{\pi}{4}\right) \\ \alpha, & \left(-\frac{\pi}{4} \leq \alpha \leq \frac{\pi}{4}\right) \\ \alpha + \frac{\pi}{2}, & \left(\alpha < -\frac{\pi}{4}\right) \end{cases} \tag{6}$$

where α is the vertical plane main direction angle.

Then at this time, there will not be two mutually orthogonal vertical main direction angles in the range of one point cloud submap. Due to errors or coarse deviations, there may be different values of the main direction angles for each vertical plane within a point cloud submap range at this time. The outlier detection method based on the median absolute deviation is used to reject the outliers for the set $\{\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_k\}$ of k $(k \geq 1)$ and vertical main direction angles within the same point cloud submap range. This method first calculates the median α_m of the k vertical plane principal direction angles and then calculates the absolute value of the difference between all the main direction angle values in the set and α_m, respectively: $\{|\alpha_1 - \alpha_m|, |\alpha_2 - \alpha_m|, |\alpha_3 - \alpha_m|, \ldots, |\alpha_k - \alpha_m|\}$. The median of these absolute values is the absolute median deviation, denoted as MAD (calculated as Equation (7)). If $\alpha_k > \alpha_m + nMAD$ or $\alpha_k < \alpha_m - nMAD$ (n is the scale factor, $n = 1$ in this paper), then α_k is determined as an outlier and is removed. The set of $m(1 \leq m \leq k)$ vertical plane main direction angles remaining after eliminating the outliers: $\{\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_m\}$ is obtained as the uniform vertical plane main direction value α_{ave} of the current point cloud submap according to Equation (8):

$$MAD = median(|\alpha_i - \alpha_m|) \tag{7}$$

$$\alpha_{ave} = \sum_{i=1}^{m} \alpha_i \tag{8}$$

where α_m is the median of the k vertical plane principal direction angles. α_i is the $i-$th vertical main direction angle. The function median means calculates the median. α_{ave} is the uniform vertical plane main direction value of the current point cloud submap. α_i is the $i-$th vertical plane main direction angle remaining after excluding the outliers.

2.2. Constructing Planar Constraints

This part corresponds to the constraint construction module shown in the dashed box in Figure 1. The main direction angle calculation method discussed in the previous section is applied to construct the planar constraints. A two-step main direction angle calculation and update strategy will be used for the vertical plane submaps to avoid possible errors introduced by unoptimized planes and to calculate the number of corrections based on the vertical submaps. The optimal correction amount based on the ground plane submap is calculated by the Gaussian–Newton method. The final six-degree of freedom correction values are used in the subsequent factor map optimization session.

2.2.1. Construction of Ground Constraints

The construction and solution process of ground constraint is shown in Figure 5. After calculating the main direction angle for the current ground point cloud submap, we first judge whether it is the first time to obtain the ground main direction angle. When the ground main direction angle is obtained for the first time, this direction angle is taken as the basis main direction angle. For the basis main direction angle, the plane parameters of the plane point cloud and the average z coordinate value Z_{ave} of all points in the plane point cloud are recorded, which is defined by Equation (9):

$$Z_{ave} = \frac{\sum_{i=1}^{n} z_i}{n} \tag{9}$$

where n is the number of points in the point cloud submap and z_i is the z coordinate value of the $i-$th point.

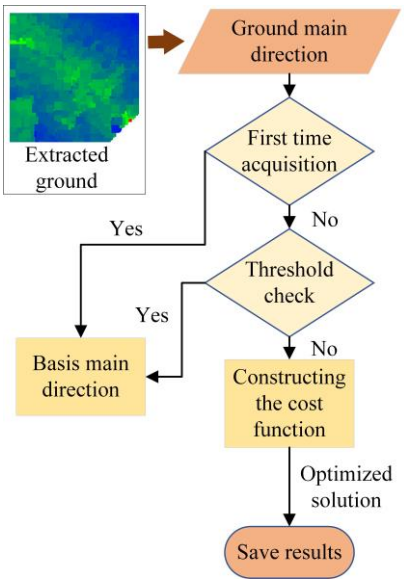


Figure 5. Flow chart of ground constraint construction. The final result is the constraint values of $roll_{opt}, pitch_{opt}, z_{opt}$ three degrees of freedom.

If the ground main direction angle is not acquired for the first time, the current main direction angle and the average z coordinate value Z_{ave} are thresholds checked. Only when the absolute value of the angle difference between the current main direction angle ang_{cur}^{ground} and the basis main direction angle ang_{basis}^{ground} is greater than Thr_{ang}^{ground} or the absolute value of the difference between the current average z coordinate value Z_{ave} and the average z coordinate value Z_{basis} of the point cloud corresponding to the basis main direction angle is greater than Thr_{z}^{ground}, the current main direction angle data is used as the new basis main direction angle.

If the threshold check is not passed, it indicates that the current ground environment has not changed significantly compared with the ground environment corresponding to the main direction angle of the basis ground. At this point, the cumulative error is eliminated by constructing and solving the cost function, using the ground coplanarity condition to constrain only three degrees of freedom of $roll$, $pitch$, and z. The cost function in this paper is shown as $F(X)$ in Equation (10). Next, the minimization problem $min\{F(X)\}$ is solved iteratively using the Gaussian–Newton method:

$$\begin{cases} F(X) = \sum_{i=1}^{n} d_g(P_i(x)) \\ d_g = \frac{|Ax_0+By_0+Cz_0+D|}{\sqrt{A^2+B^2+C^2}} \\ P_i(x) = R[x,y,z]^T + [0,0,z_{opt}]^T \end{cases} \tag{10}$$

where $X = [roll, pitch, z_{opt}]^T$ is the vector composed of the state volume to be estimated, $x_i = [x,y,z]^T$ is the $i-$th point (n points in total) in the current point cloud, which is $x = [x_0, y_0, z_0]^T$ after the transformation using the state volume to be estimated, d_g is the distance from (x_0, y_0, z_0) to the plane corresponding to the main direction angle of the basis ground (Equation (2)), $P_i(X)$ is the transformation relationship before and after each solution, and R is the rotation matrix obtained by the transformation of $roll_{opt}$, $pitch_{opt}$.

Finally, if it converges successfully, the solution result $[roll_{opt}, pitch_{opt}, z_{opt}]^T$ will be saved, and if it does not converge successfully, this constraint will be skipped, and the solution result will not be saved.

2.2.2. Construction of Vertical Planar Constraints

A two-step vertical plane main direction constraint algorithm shown in Figure 6 is used. In the first stage, the historical point cloud of k frames before n frames are first taken to form a submap of the vertical plane point cloud for the calculation of the main direction angle of the vertical plane. Since the selected historical point clouds are some time away from the current frame, the constraint and optimization of these frames have already been optimized. By adopting such a lag update strategy for the vertical plane main direction angle, the possible errors caused by introducing unoptimized planes can be avoided. Next, the vertical plane submaps are obtained, and the vertical plane main directions are calculated using the method in the previous section. If this is the first time obtaining the vertical plane main direction angle $ang_i^{history}$ of the historical point cloud submap, this main direction angle will be added to the basis vertical plane main direction angle set $AngSet$, which is defined as shown in Equation (11):

$$AngSet = \left\{ ang_i^{history} \right\} \tag{11}$$

where $ang_i^{history}$ is the vertical plane main direction angle of the historical frame point cloud submap acquired for the $i-$th time.

Use the MAD method mentioned in Section 2.1.3 for $AngSet$ to remove the outliers and average them as the basis main direction angle $ang_{basis}^{vertical}$. Otherwise, this angle is checked, and if the absolute value of the difference between the current main direction angle $ang_{history}^{vertical}$ and the basis main direction angle $ang_{basis}^{vertical}$ is greater than $Thr_{ang}^{vertical}$, then this main direction angle is added to the set $AngSet$ of the main direction angle of the basis vertical plane.

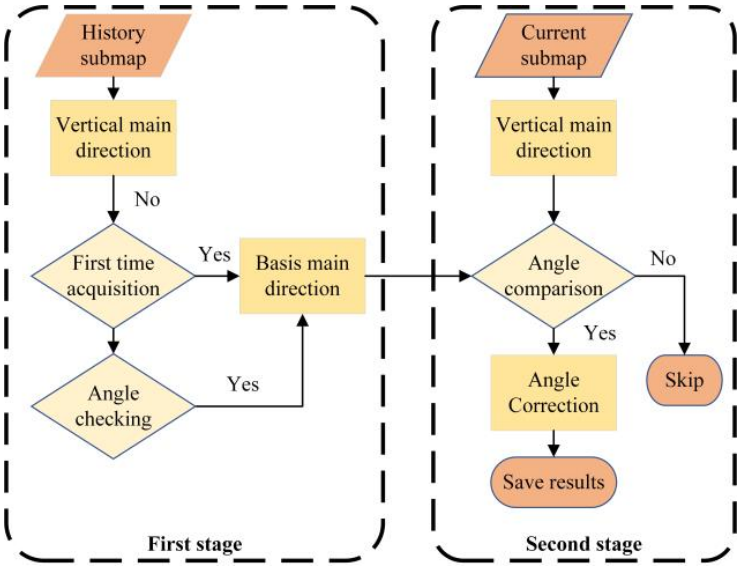


Figure 6. Flow chart of the vertical planar constraint construction. The final result is the constraint values for the three degrees of freedom yaw, x_{opt}, y_{opt}.

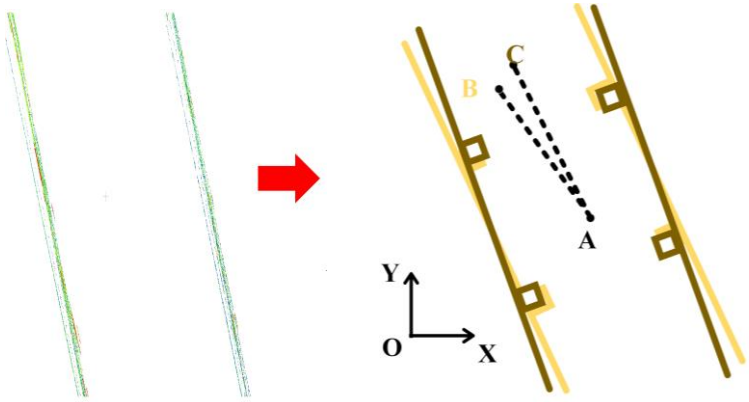
In the second stage, the above method calculates the vertical plane's main direction angle composed of the current k-frames point cloud for the vertical plane submap. Next, the current vertical plane main direction angle $ang_{current}^{vertical}$ is compared with the basis vertical plane main direction angle $ang_{basis}^{vertical}$. If the absolute value of the difference between the two angles is greater than $Thr_{correct}^{vertical}$, it is considered that the current threshold condition is not satisfied and the difference of the main direction angle is not within a reasonable range, so the current submap is subsequently not processed to avoid the introduction of wrong constraints.

When the threshold condition is satisfied, it is considered that the difference in the main direction angle of the current vertical plane submap is within a reasonable range compared with the historical submap. The constraint is constructed and solved for the vertical plane submap and based on the main direction angle of the vertical plane, the three degrees of freedom of yaw, x, y can be constrained. The diagram of constraint construction is shown in Figure 7, $A(x_A, y_A)$ is the horizontal position corresponding to the center frame of the vertical plane submap of the historical frame, $B(x_B, y_B)$ is the horizontal position corresponding to the center frame of the current vertical plane submap, and $C(x_C, y_C)$ is the new horizontal position of the center frame of the current vertical plane submap after correcting the main direction angle of the current vertical plane with the main direction angle of the basis vertical plane.

$$angle_{correct} = \arctan\frac{y_A - y_B}{x_A - x_B} + ang_{basis}^{vertical} - ang_{current}^{vertical} \tag{12}$$

$$\begin{cases} x_C = x_A + sqrt\left((x_A - y_A)^2 + (x_B - y_B)^2\right) * cos(angle_{correct}) \\ y_C = y_A + sqrt\left((x_A - y_A)^2 + (x_B - y_B)^2\right) * sin(angle_{correct}) \end{cases} \tag{13}$$

where $ang_{basis}^{vertical}$ is the basis main direction angle and $ang_{current}^{vertical}$ is the current main direction. The $angle_{correct}$ is the main direction angle of the current vertical plane after correction.



(**a**) Extracted planes (Top view) (**b**) Constraint construction

Figure 7. Schematic diagram of constraint construction for vertical plane submap: (**a**) shows the point cloud of the historical vertical plane extracted from a section of the corridor displayed with the current vertical plane superimposed; (**b**) shows the schematic diagram of constraint construction under the scene in (**a**).

The $C(x_C, y_C)$ is calculated as shown in Equation (13), where the angle $angle_{correct}$ of the main direction of the current vertical plane after making corrections is calculated according to Equation (12). After getting the horizontal coordinates of C, the constraint values for the three degrees of freedom: $yaw_{opt}, x_{opt}, y_{opt}$ can be calculated as shown in Equation (14). The result of the calculation will be saved as $\left[yaw_{opt}, x_{opt}, y_{opt}\right]^T$:

$$\begin{cases} yaw_{opt} = ang_{basis}^{vertical} - ang_{current}^{vertical} \\ x_{opt} = x_C - x_B \\ y_{opt} = y_C - y_B \end{cases} \tag{14}$$

where $yaw_{opt}, x_{opt}, y_{opt}$ is the constraint value of the three degrees of freedom.

2.3. Factor Graph Optimization

To solve the drift caused by the error accumulation in the system operation, this paper uses the six-degree of freedom planar constraint value $\{roll_{opt}, pitch_{opt}, yaw_{opt}, x_{opt}, y_{opt}, z_{opt}\}$ calculated in the previous subsection to provide inter-frame constraints on the estimated poses of the system. Based on the existing back-end factor graph optimization framework of the LIO-SAM algorithm, the state of the system is treated as the quantity to be optimized [4] and is added to the factor graph in the form of variable nodes. In this paper, planar constraints are added between variable nodes in the form of factor nodes. Meanwhile, other constraints in the LIO-SAM system, such as loop closure constraints and LiDAR inter-frame constraints, are added to the factor graph framework for optimization together with other constraints. The optimization is performed using incremental smoothing with Bayesian tree (iSAM2) mapping [26] whenever a new node is added to the factor graph. The optimization is performed in a sliding window to ensure efficiency. A globally consistent pose can be obtained by solving the optimal estimation of individual variable nodes. The final optimized global point cloud map can be obtained using this pose to accumulate the point cloud in the global coordinate system.

3. Experiment

In this section, we have collected datasets in various environments, including indoor and outdoor, and tested our method, with four algorithms, LeGO-LOAM, LIO-SAM, and FAST-LIO2, using these datasets for comparison. To adapt to the backpack laser scanning system used in this paper, the data input parts of each type of algorithm are modified accordingly. These modifications do not cause differences in the results of the

four algorithms. In the previous section, some parameters included in the method of this paper are introduced, and the values of these parameters need to be selected according to the data scenarios. The exact parameter values are shown in Table 1. In the experimental scenarios of this paper, the parameters in Table 1 apply to each scenario. These parameters were obtained by the trial-and-error method. Since the parameters for the same function in different algorithms took the same values in this paper's qualitative and quantitative experiments, the choice of parameters does not affect the experimental results. The default values are used for each parameter of other algorithms.

Table 1. Table of parameter values.

Variable	Value
k	10
Thr^{filter}(Ground plane)	(1,1,1.5)
Thr^{filter}(Vertical plane)	(20,20,0.5)
Thr_{MAE}	0.01
Thr_n	10
Thr_{ang}^{ground}	1.5
Thr_z^{ground}	0.2
n	200
$Thr_{ang}^{vertical}$	1
$Thr_{correct}^{vertical}$	2

3.1. Data Description

To verify the effectiveness of the method proposed in this paper, the raw data were collected in three real environments: an outdoor campus, an indoor corridor, and an underground parking, lot using the backpack laser scanning system shown in Figure 8. The device contains two sixteen-line LiDARs, an IMU with a frequency of 200 Hz, and a GNSS receiver. The calibration has been done manually between the LiDAR, IMU, and GNSS receiver.

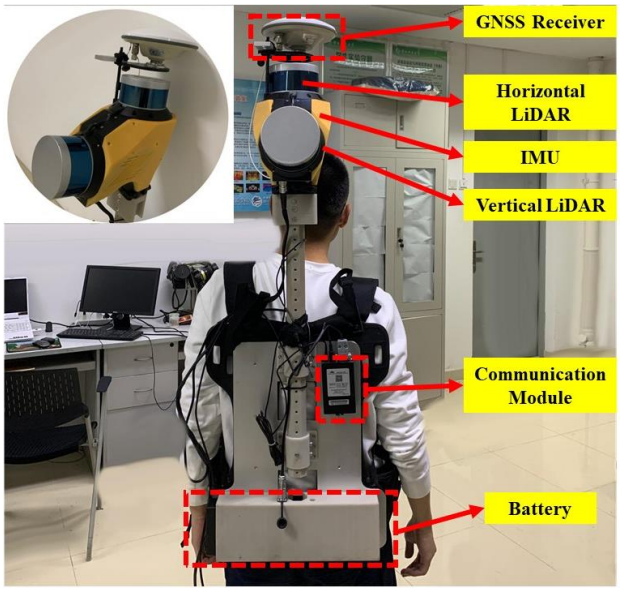


Figure 8. Backpack laser scanning system. Includes two Velodyne VLP-16 LiDARs (horizontally and tilt mounted, respectively), built-in imu at 200 Hz, GNSS receiver, a communication module for receiving GNSS differential data, and battery.

The experimental data set (shown in Figure 9) consists of three sets of data collected in the three different environments mentioned above. Figure 9a shows the data collected in an

above-ground indoor environment containing a straight corridor of approximately 70 m in length and a manually leveled floor. Figure 9b shows the data collected in an underground parking lot containing vehicles, columns, walls, and artificially leveled floors. This paper considers the corridor and underground parking lot datasets as two typical environments that conform to the MW assumption. We will test the performance of our method in these two datasets. The data shown in Figure 9c are collected in the campus environment, which is not considered a typical environment in conforming to the MW assumption because it contains more structures that do not conform to the MW assumption, such as vehicles, trees, streetlights, etc. We hope to test the impact of the non- MW assumption environment on the method in this dataset. The acquisition time, trajectory length, and the maximum change of ground elevation during the acquisition for the three datasets are shown in Table 2. Since the GNSS signals in some sections of the campus dataset were obscured by trees and so on, and the true value data with sufficient accuracy could not be obtained, three sub-datasets with good GNSS signals were selected in the campus dataset, and the details of the three sub-datasets are shown in Table 3.

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

Figure 9. Experimental dataset real-world images: (**a**) is an actual image of the corridor dataset; (**b**) is an actual image of the parking lot dataset; (**c**) is an actual image of the campus dataset.

Table 2. Data set details.

Dataset	Duration (s)	Trajectory Length (m)	Elevation Change (m)
Corridor	657.7	412.89	0
Underground Parking	774.5	621.109	0
Campus	950.0	935.088	2.26

Table 3. Sequence details of the campus dataset.

Dataset	Timestamp (s)	Duration (s)
Campus-01	105,981.6~106,180.1	198.5
Campus-02	106,480.1~106,650.1	170.0
Campus-03	106,850.1~106,929.1	79.0

In acquiring the three datasets, a segment was acquired in the area with an excellent outdoor GNSS signal. The GNSS trajectory corresponding to this SLAM output trajectory was obtained (the GNSS data were differentially processed to improve the accuracy) so that the conversion relationship between the SLAM global coordinate system and the WGS84 coordinate system provided by GNSS could be derived using these two segments. Therefore, point cloud maps in the WGS84 coordinate system can be obtained for all three experimental datasets. The ground part of the data used to derive the trajectory conversion relationship has been manually cut out for the corridor and parking lot datasets. This part of the data is not discussed in the experimental part. The experimental data were processed on a computer with an Intel Core i5-11400H @ 2.7 GHz processor and 16 GB RAM.

3.2. Evaluation Indicators

To compare the performance of the two algorithms in real environments, the reflective markers shown in Figure 10 are placed in two GNSS-denied environments, the corridor, and the underground parking lot. The significant laser reflectivity of the reflective markers allows us to find them from the point cloud map using the intensity view of the point cloud. The reflective markers have been placed as evenly and adequately as possible in the GNSS-denied environment. We obtained the absolute coordinates of these reflective markers in the WGS84 coordinate system in three axes by mapping methods using a total station and other measuring instruments. In the quantitative experiments, this coordinate value is used as the true value to evaluate the accuracy of these two indoor scenes.

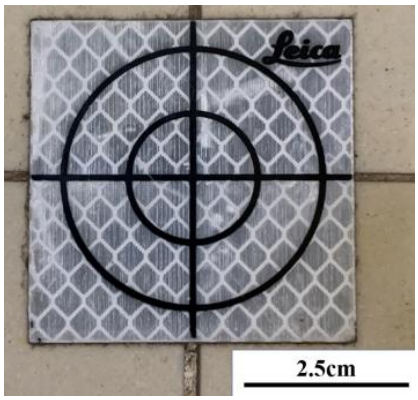


Figure 10. Reflective markers. The coating on the surface of this sign gives it a greater laser reflectivity, enabling it to be identified in the point cloud map by its intensity.

This paper uses a backpack laser scanning system for outdoor scenes to obtain the GNSS data. It gets more accurate trajectory coordinate data by surveying and mapping post-processing with the data output from an additionally placed GNSS reference station. These trajectory coordinate data are used as the true value to evaluate the accuracy of the outdoor scenes in the quantitative experiments. Since only three sub-datasets of the campus dataset have good GNSS coverage to obtain the true values, the quantitative evaluation of the campus dataset is performed only for these three sub-datasets. Furthermore, trajectory plots, as well as error distribution plots, were performed using the EVO evaluation tool [27].

This paper uses the root mean square error (RMSE) for quantitative experiments as an accuracy evaluation metric and measures the algorithm performance by this metric. In this paper, RMSE represents the error between the coordinates obtained by manually selecting the center of the ground marker point cloud and measuring the true value in the point cloud map output by SLAM. For the $i-$th reflective marker, if its coordinates in the point cloud map are $\left(x_i^m, y_i^m, z_i^m\right)$, and the corresponding true value is $\left(x_i^t, y_i^t, z_i^t\right)$, then the RMSE is calculated as shown in Equation (15):

$$RMSE = \sqrt{\frac{1}{num_t}\sum_1^{num_t}\left[\left(x_i^m - x_i^t\right)^2 + \left(y_i^m - y_i^t\right)^2 + \left(z_i^m - z_i^t\right)^2\right]} \tag{15}$$

where num_t is the total number of reflective marks placed in this scene.

3.3. Qualitative Experiments

This paper analyzes three datasets using our method and three other algorithms. In this subsection, the proposed method will be evaluated qualitatively from two aspects: the evaluation of the vertical plane extraction effect and the evaluation of the ground constraint effect. The effectiveness of our method is evaluated by comparing the trajectory analysis with other methods. The point clouds obtained using our method for the three

datasets are shown in Appendix A, and the point clouds are assigned according to the Z coordinate values.

3.3.1. Vertical Plane Extraction Evaluation

The results of vertical plane extraction for the three datasets using the method proposed in this paper are shown in Figure 11, where the point clouds are color assigned according to the intensity values. Results (a–d) show the elevation extraction results for the corridor and the underground parking lot datasets. Combining the top view and front view, we can see that for the corridor and parking lot, which conforms with the assumption of the MW, the vertical plane extraction method in this paper can extract the vertical planes in the scene more completely and can reflect the general outline of the scene. The extracted planes have no floor or ceiling, so the extraction accuracy is good. The red boxes mark some structures within the two types of scenes that do not conform to the MW assumption. Thanks to this paper's statistical-based principal orientation angle extraction strategy, even if a few structures within the scenes do not conform to the assumptions, they do not significantly impact the basis main direction angle values. Since the campus dataset is not a typical environment conforming to the MW assumption, only a small number of vertical planes are extracted in this dataset. Results (e) and (f) show the partial elevation extraction results of this dataset. In the environment that does not conform to the MW assumption, the facade constraint module of this paper only works in the few cases that conform to the threshold set by the method of this paper and do not affect the system in the rest of the cases.

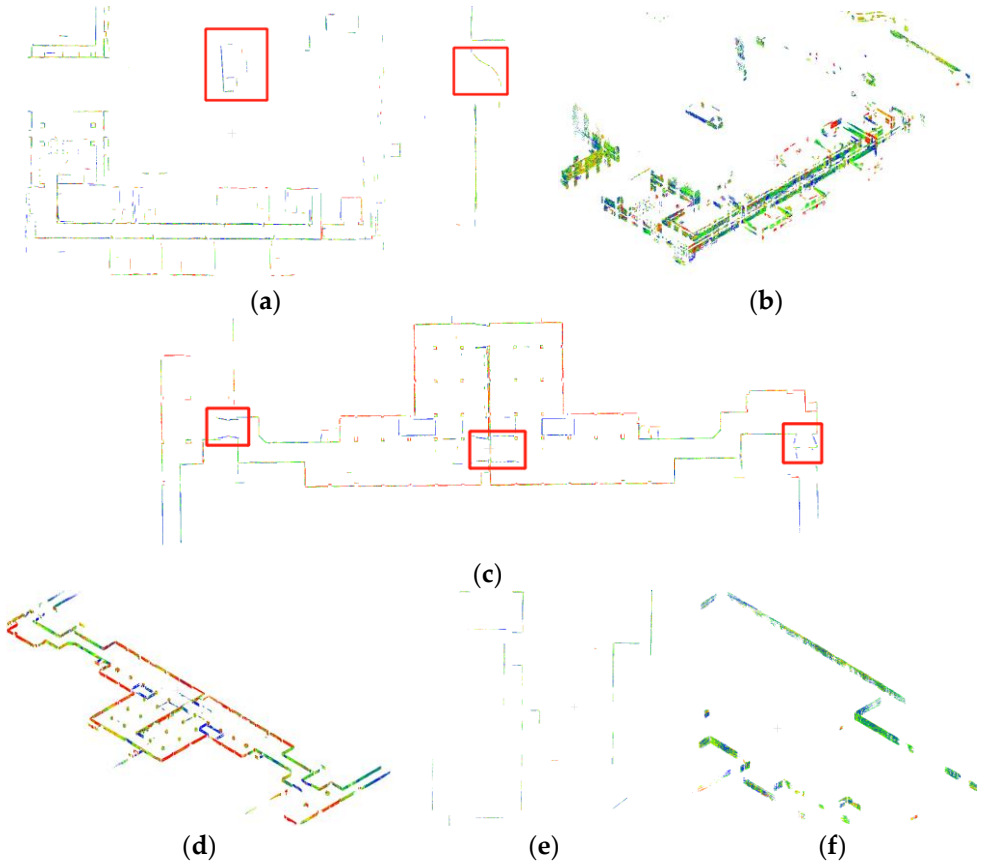
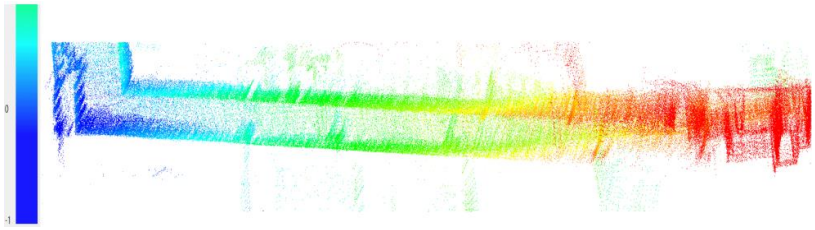


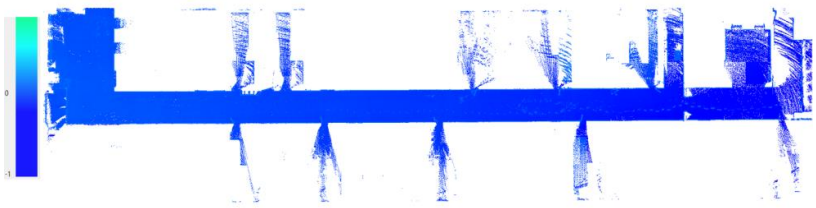
Figure 11. The results of vertical plane extraction for the three datasets: (**a**,**b**) are the results of corridor dataset extraction, (**c**,**d**) are the results of parking lot dataset extraction, and (**e**,**f**) are the results of campus dataset extraction. (**a**,**c**,**e**) are the top view, (**b**,**d**,**f**) are the front view.

3.3.2. Ground Constraint Evaluation

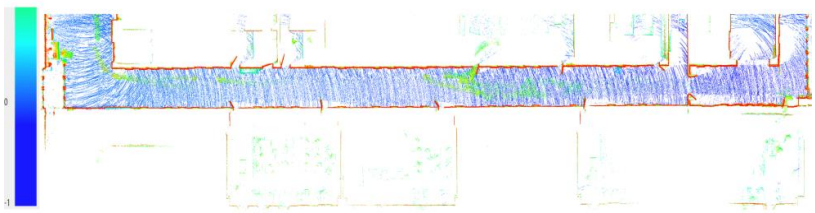
After ground constraining the three datasets using the method in this paper, the ground point clouds obtained are compared to the ground point clouds obtained through the other algorithms in terms of elevation, as shown in Figure 12, and the point clouds are color assigned according to the elevation values (*Z* coordinate values). Although the LeGO-LOAM algorithm constrains the ground, it is designed explicitly for ground-based unmanned vehicles. It requires the LiDAR to be installed at a position approximately parallel to the ground or to convert its data to be approximately parallel to the ground. Our backpack laser scanning system does not meet the requirements of this algorithm well, so the LeGO-LOAM algorithm performs poorly in all three dataset species. In the corridor dataset, due to the small environmental elevation change, the ground point clouds (b), (c), and (d) obtained by the other methods do not show significantly different elevation trends, except for the LeGO-LOAM result (a), which shows a significant ground elevation change. However, the blue color from left to right in (b) and (c) gradually deepens, representing the gradual decrease of ground elevation. The color distribution in (d) obtained by our method is more uniform and consistent with the actual situation of no change in the corridor elevation. In the underground parking lot dataset, there is no significant change in the ground elevation in the actual environment. The color distribution of the ground point cloud shown in (g) obtained by our method is uniform and consistent with the actual elevation change. The point clouds in (e) and (f) have a significant elevation change from left to right, and the point cloud on the left is significantly lower than the point cloud on the right and in the middle. In this dataset, the LeGO-LOAM algorithm cannot obtain typical point clouds. In the campus dataset, the actual environment has ground height undulations, and the results (k), (i), and (j) obtained by our method reflect the real environmental elevation changes to some extent, and (h) shows a noticeable point cloud distortion.



(**a**) LeGO-LOAM

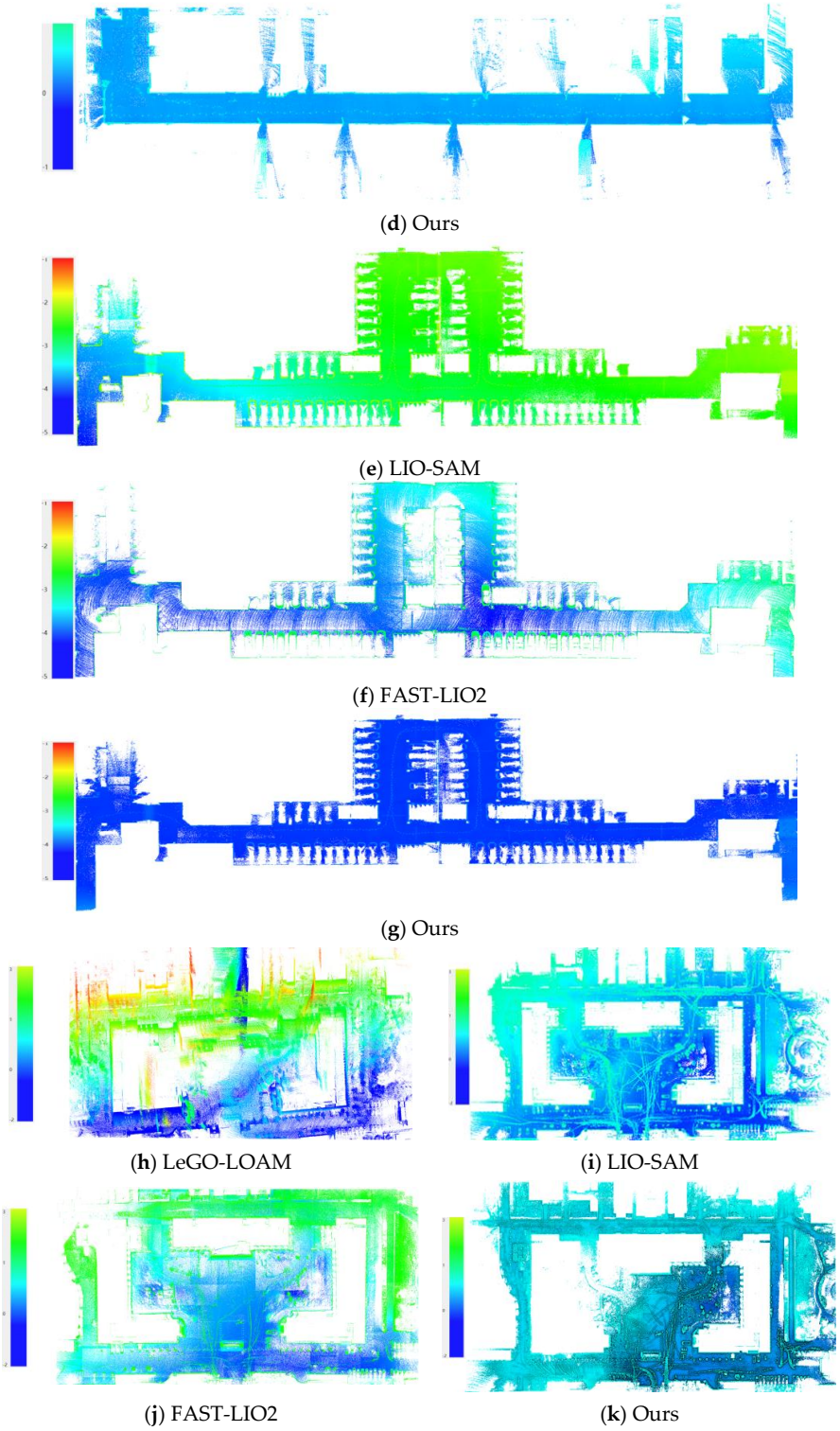


(**b**) LIO-SAM



(**c**) FAST-LIO2

Figure 12. *Cont.*



(**d**) Ours

(**e**) LIO-SAM

(**f**) FAST-LIO2

(**g**) Ours

(**h**) LeGO-LOAM

(**i**) LIO-SAM

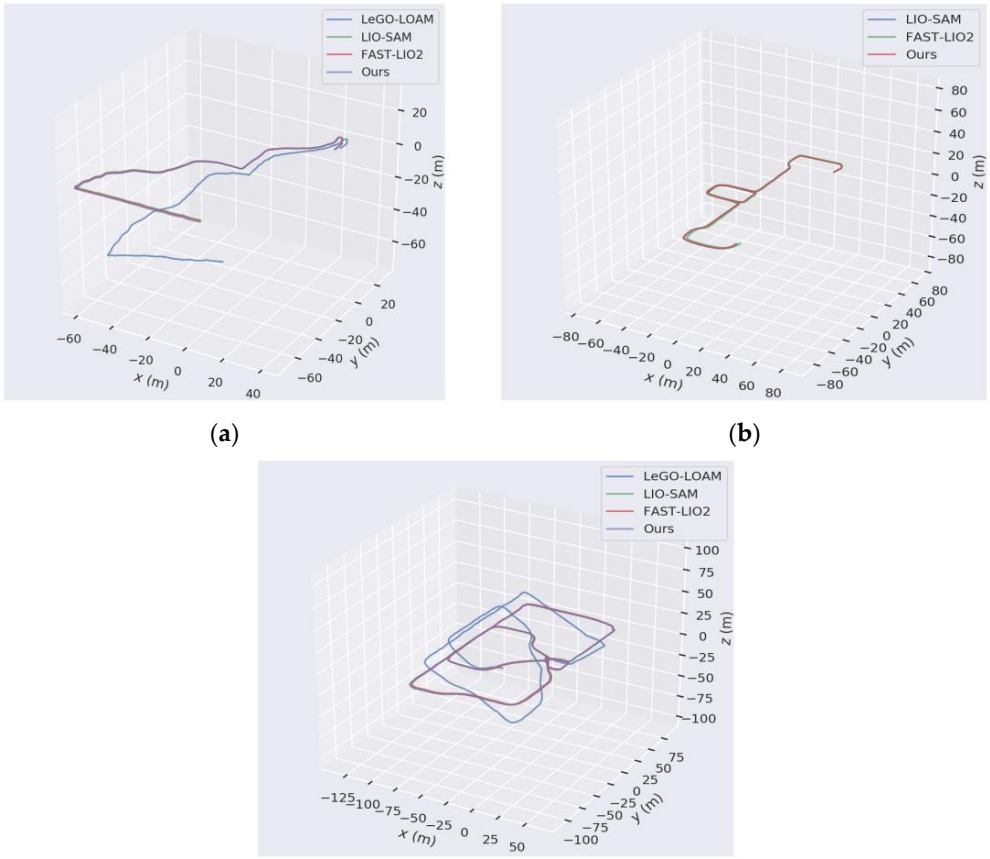
(**j**) FAST-LIO2

(**k**) Ours

Figure 12. Comparison of ground elevation of three datasets: (**a**–**d**) are the corridor dataset, (**e**–**g**) are the underground parking lot dataset, and (**h**–**k**) are the campus dataset.

The trajectories and trajectory coordinates obtained using the method in this paper and the other methods are shown in Figures 13 and 14. Because the trajectories obtained by LeGO-LOAM and the other three methods are too different, we do not depict the trajectory coordinates of the LeGO-LOAM algorithm in Figure 14 to better describe the trajectory details of the other three algorithms. It can be seen more clearly from the trajectory variations that the trajectories obtained by our method better reflect the elevation variations

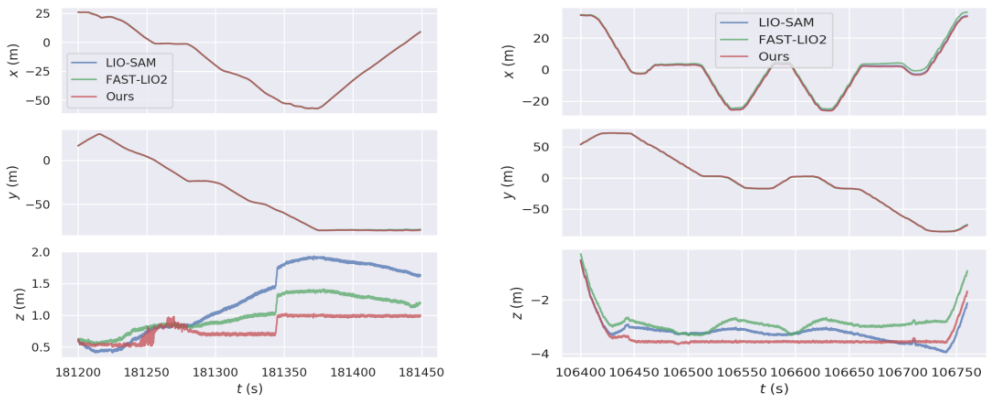
of the real environment in the corridor dataset and the underground parking lot dataset. In the campus environment, the elevation change trend of our method is similar to that of LIO-SAM and FAST-LIO2 at the significant elevation change. It only plays the role of ground constraint at the slight elevation undulation. The elevations of the LIO-SAM and FAST-LIO2 trajectories differ significantly from those of the present method near the time stamp 106,400 s. We will further explore this issue in our quantitative analysis.



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

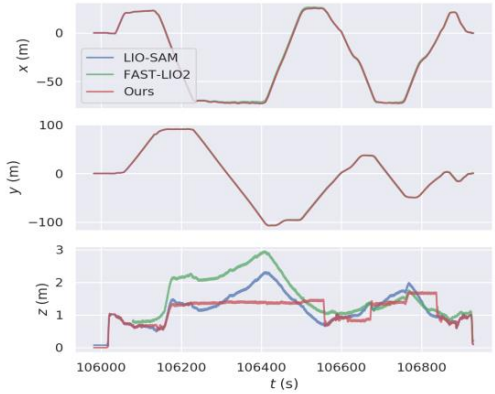
(**c**)

Figure 13. Three datasets of various methods of trajectory comparison: (**a**) is the trajectory comparison of the corridor dataset; (**b**) is the trajectory comparison of the underground parking dataset; (**c**) is the trajectory comparison of the campus dataset.



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

Figure 14. *Cont.*



(**c**)

Figure 14. Comparison of trajectory coordinates of various methods for three datasets: (**a**) is the comparison of trajectory coordinates of the corridor dataset; (**b**) is the comparison of trajectory coordinates of the underground parking dataset; (**c**) is the comparison of trajectory coordinates of the campus dataset.

Combined with the above analysis, in structured environments such as corridors and parking lots, the ground constraint method in this paper can better reflect the elevation changes of the real environment. In unstructured environments such as campuses, the ground constraint method in this paper will only work in those parts that conform to the MW assumption.

3.4. Quantitative Experiment

For the corridor dataset and the underground parking lot dataset, we use Equation (15) to calculate the RMSE values of the reflective marks in the point clouds as a metric to quantitatively evaluate the quality of the point cloud maps for these two datasets. To more clearly reflect the method's performance in this paper, we calculate the RMSE of the reflected signs in the X, Y, and Z directions, respectively, and also calculate the RMSE of the coordinates to describe the overall error.

Although the corridor and underground parking lot datasets included a small portion of the above-ground environment at the time of acquisition, our quantitative evaluation focused only on the indoor part of these two datasets. The RMSEs of the reflection signs for the four methods are shown in Table 4, with "/" indicating that valid results could not be obtained. There is no GNSS signal indoors or a lack of constraint, so the accuracy of the three methods, LeGO-LOAM, LIO-SAM, and FAST-LIO2, is poor, especially the error in the z-direction, which is significant. In this paper, except for the x-direction of the corridor dataset, the method achieves optimal accuracy in other directions of both datasets and the overall error. The accuracy is greatly improved compared with the other three methods. Compared with the LeGO-LOAM, LIO-SAM, and FAST-LIO2 methods, the overall errors of this paper's method are reduced by 99.5%, 73.4%, and 93.4% in the corridor data set, and 41.4% and 79.6% in the underground parking lot data set, respectively. Therefore, the method in this paper has high accuracy in the environment that conforms to the MW assumption.

For the campus dataset, we calculate the RMSE of the trajectories obtained by the other three types of methods and the trajectories obtained by this paper relative to the ground truth in each of the three sub-datasets with good GNSS coverage (as shown in Table 5), and then plot the distribution of the errors in the trajectories as shown in Figure 15. The accuracy of this paper's method on the Campus-01 dataset and the Campus-03 dataset differs less from the optimal results in an environment such as the campus dataset, which does not conform to the MW assumption. The RMSE of this method increases by 1.1 cm in the Campus-01 dataset and by 0.9 cm in the Campus-03 dataset compared to the optimal method, while the RMSE of this method increases by 9.2 cm in the Campus-02 dataset.

The decrease in accuracy may be due to the error optimization caused by the fact that our parameter settings are the same throughout the entire dataset, in conjunction with our discussion of the significant difference between the point cloud elevations obtained by this method and other methods near the time stamp 106,400 s in our qualitative experiments. We will discuss this issue further in Section 4.

Table 4. Corridor dataset and underground parking dataset point cloud reflection sign RMSE(m).

Dataset	Indicator	LeGO-LOAM	LIO-SAM	FAST-LIO2	Ours
Corridor	X-axis error	11.123	0.046	0.113	0.084
	Y-axis error	8.522	0.268	0.238	0.115
	Z-axis error	30.784	0.567	2.503	0.087
	Overall error	33.823	0.629	2.516	0.167
Underground Parking	X-axis error	/	0.239	1.187	0.123
	Y-axis error	/	0.149	0.202	0.064
	Z-axis error	/	0.417	0.796	0.260
	Overall error	/	0.503	1.443	0.295

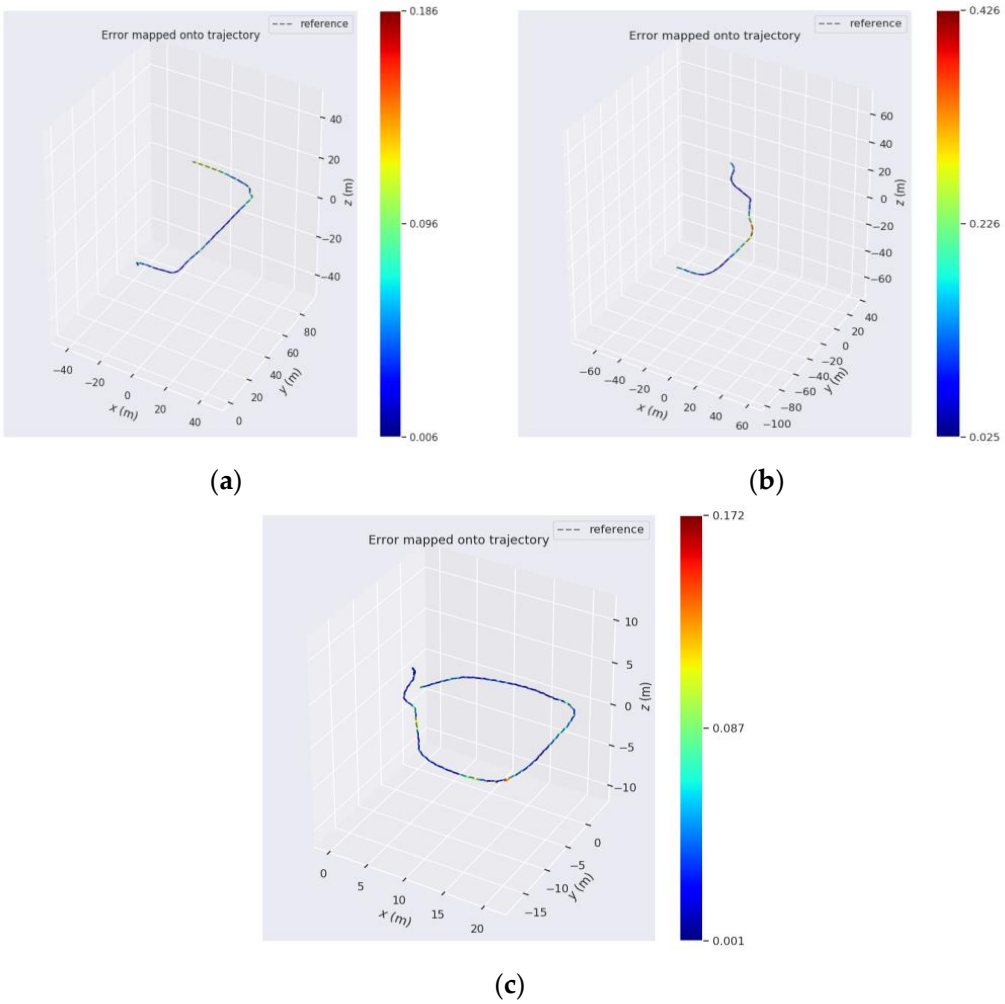


Figure 15. The distribution of the errors in the trajectories obtained by the method in this paper compared with the ground truth in the three sub-datasets of the campus dataset is plotted in the trajectories. Where (**a**) corresponds to sub-dataset Campus-01, (**b**) corresponds to sub-dataset Campus-02, (**c**) corresponds to sub-dataset Campus-03.

Table 5. Trajectory RMSE errors (m) for the campus dataset.

Dataset	LeGO-LOAM	LIO-SAM	FAST-LIO2	Ours
Campus-01	0.921	**0.048**	0.058	0.059
Campus-02	0.205	0.090	**0.067**	0.159
Campus-03	/	0.052	**0.036**	0.045

4. Discussion

This paper hopes to introduce additional constraints to the system to enhance the stability and robustness of the LiDAR SLAM system. Based on this, we use the MW assumption to find the parallel–perpendicular relationship between the planes of man-made buildings and introduce it into the LiDAR SLAM system as an a priori constraint. We believe that this work can be applied and can improve the accuracy in many scenarios such as parking lots, indoor rooms, long corridors, etc. Although there have been some planar constraint algorithms for vision-based SLAM, this effective idea of introducing additional constraints in the field of LiDAR SLAM has not been widely studied.

The method in this paper may be mis-extracted when performing vertical plane extraction. There may be a small number of structures in the scene that do not conform to the MW assumption and thus, are susceptible to interference from these coarse differences in the main direction angle calculation. To eliminate these coarse differences, we adopt a statistical-based two-step principal direction angle calculation and an updating strategy for the elevations to improve the stability and robustness of the system. The possible errors introduced by the unoptimized planes are avoided by lagging the update of the main direction angle. The setting of parameters affects the performance of our method. For scenarios that conform to the assumptions, the structured features, such as the parallel perpendicularity of the planes, are more obvious, and there is no need to change the parameters in the process. However, for environments that do not conform to the MW assumption, it may be necessary to use different parameters at different locations within a dataset to cope with the environmental changes, as the environment is more diverse. Therefore, in the experiments for the campus scenario in Section 3.3, there will be an increase in trajectory for the errors in the Campus-02 dataset, which may be caused by inappropriate parameter settings, such as an overly large update thresholds for the ground main direction angle. Too large an update threshold will reduce the ability of the method in this paper to describe the real environment terrain. Still, too small a ground main direction angle update threshold will have frequent main direction angle updates. When the main direction angle is updated, it is usually in some undulating terrain area. Since the old ground main direction angle no longer applies to the current environment, only the ground main direction angle can be updated, and no constraints can be added. Therefore, frequent main direction angle updates will reduce the constraints.

In the previous paper, we did not discuss the algorithm's time efficiency due to the significant time consumption in planar extraction, so the method in this paper cannot achieve real-time performance yet. Currently, the method in this paper still needs 1~1.5 times the acquisition time to complete the calculation.

5. Conclusions

In this paper, a planar constraint-assisted LiDAR SLAM algorithm based on the MW assumption is proposed. Planes are extracted based on the priori knowledge of parallel–perpendicular relationships between planes in the environment. Main direction angles are computed to provide additional constraints for a SLAM state estimation. A statistically based planar main direction angle update strategy is designed to enhance robustness to the small number of structures in the scene that do not conform to the MW assumption. We have collected data and completed qualitative and quantitative experiments using a backpack laser scanning system in various indoor and outdoor environments. We also investigate the performance of our method in a non-MW assumption environment. The

results show that our method can improve the accuracy in environments that conform to the MW assumption and has good stability and robustness for environments with a small number of unstructured objects.

In the subsequent work, on the one hand, we will aim to make fuller use of the planar features in the scene to improve the front-end odometer's performance and reduce the system's cumulative error. We will also simplify the plane extraction process and remove the redundant planar constraints to improve the real-time performance of the system. On the other hand, we will conduct a more in-depth study on the environments that do not conform to the MW assumption to find more general and robust planar constraint methods.

Author Contributions: H.W. and W.W. carried out the empirical studies, the literature review, and drafted the manuscript; X.Q. and C.W. assisted with the data collection; R.Z. and L.A. helped to draft and review the manuscript, and communicated with the editor of the journal. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Key Technologies Research and Development Program of China under Grant 2022YFB3904101, in part by the National Natural Science Foundation of China under Grant U22A20568 and 42071444.

Data Availability Statement: Not applicable.

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

Appendix A

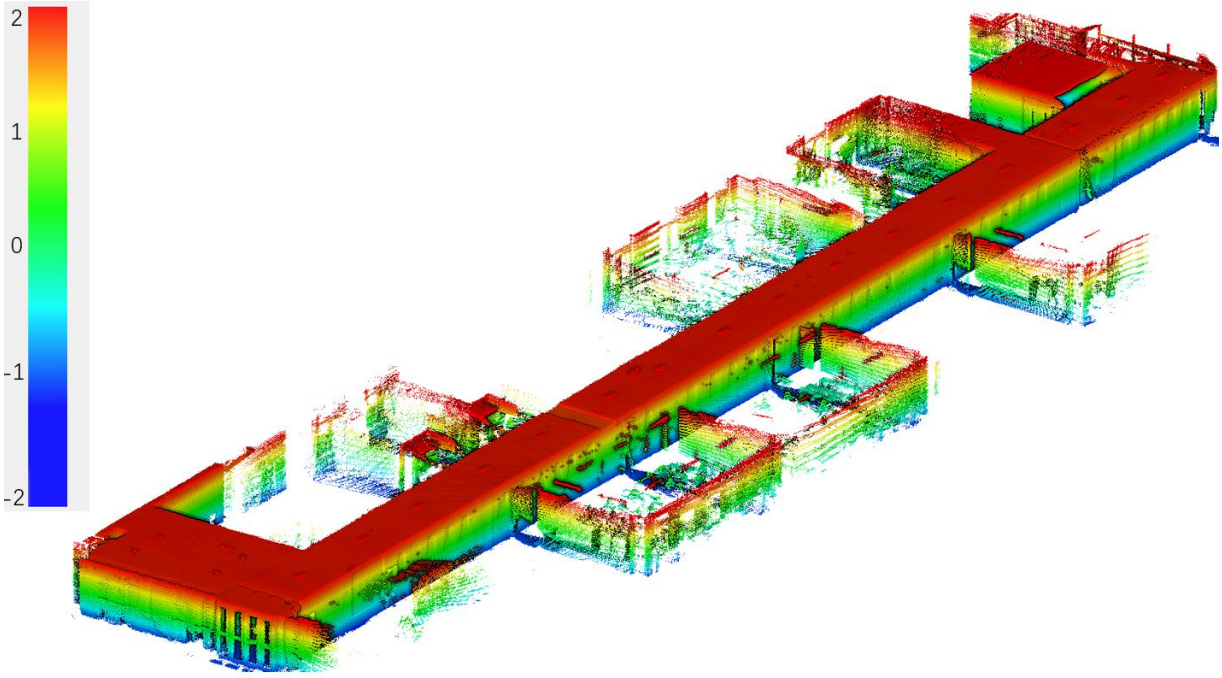


Figure A1. Point cloud obtained from the corridor dataset.

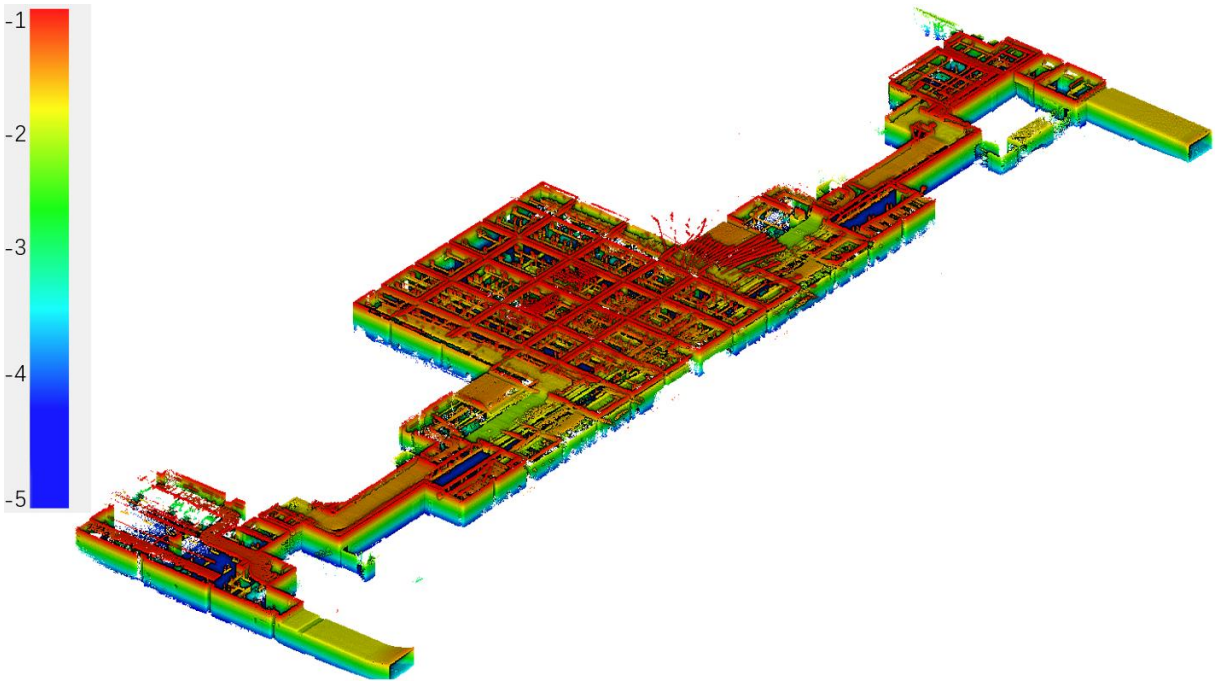


Figure A2. Point cloud obtained from the underground parking lot dataset.

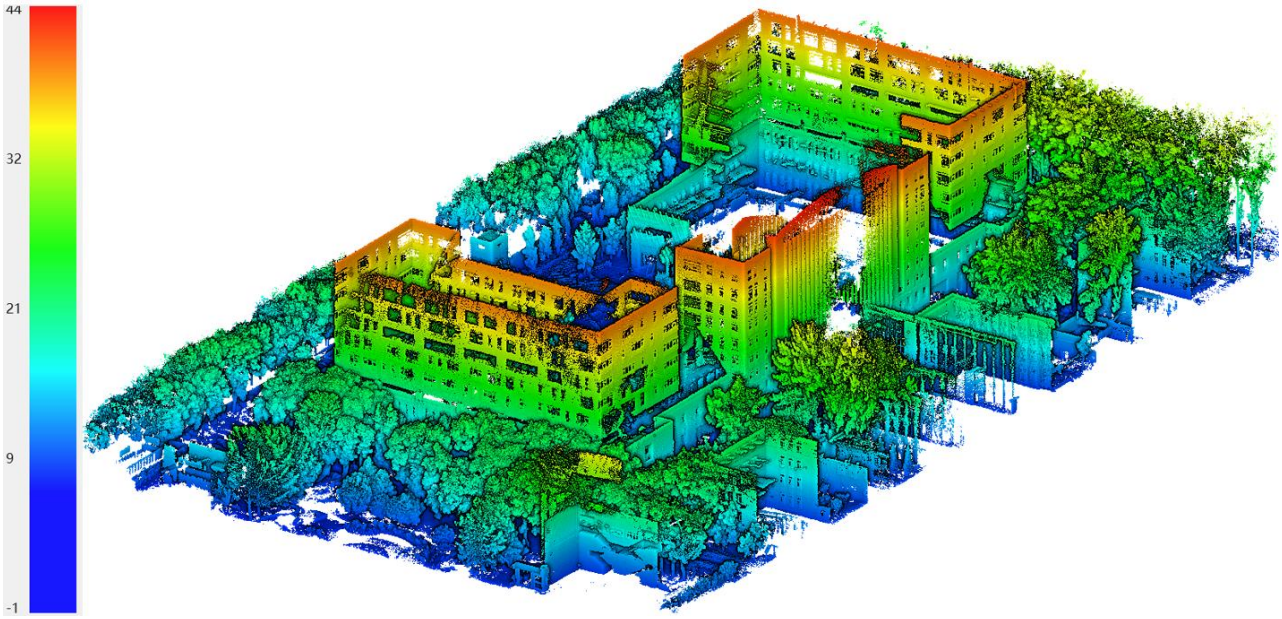


Figure A3. Point cloud obtained from the campus dataset.

References

1. Qin, T.; Li, P.; Shen, S. VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator. *IEEE Trans. Robot.* **2018**, *34*, 1004–1020. [CrossRef]
2. Campos, C.; Elvira, R.; Rodríguez, J.J.G.; Montiel, J.M.M.; Tardós, J.D. ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual–Inertial, and Multimap SLAM. *IEEE Trans. Robot.* **2021**, *37*, 1874–1890. [CrossRef]
3. Ji, Z.; Singh, S. LOAM: Lidar Odometry and Mapping in Real-Time. In Proceedings of the Robotics: Science and Systems Conference (RSS), Berkeley, CA, USA, 12–14 July 2014.
4. Shan, T.; Englot, B.; Meyers, D.; Wang, W.; Ratti, C.; Rus, D. LIO-SAM: Tightly-Coupled Lidar Inertial Odometry via Smoothing and Mapping. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 24 October 2020–24 January 2021; pp. 5135–5142.

5. Kim, G.; Kim, A.; Kosecka, J. 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.
6. Xu, W.; Cai, Y.; He, D.; Lin, J.; Zhang, F. Fast-lio2: Fast direct lidar-inertial odometry. *IEEE Trans. Robot.* **2022**, *38*, 2053–2073. [CrossRef]
7. Shan, T.; Englot, B. LeGO-LOAM: Lightweight and Ground-Optimized Lidar Odometry and Mapping on Variable Terrain. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 4758–4765.
8. Gomez-Ojeda, R.; Moreno, F.A.; Zuñiga-Noël, D.; Scaramuzza, D.; Gonzalez-Jimenez, J. PL-SLAM: A Stereo SLAM System through the Combination of Points and Line Segments. *IEEE Trans. Robot.* **2019**, *35*, 734–746. [CrossRef]
9. Zhang, X.; Wang, W.; Qi, X.; Liao, Z.; Wei, R. Point-Plane SLAM Using Supposed Planes for Indoor Environments. *Sensors* **2019**, *19*, 3795. [CrossRef] [PubMed]
10. Geneva, P.; Eckenhoff, K.; Yang, Y. LIPS: LiDAR-Inertial 3D Plane SLAM. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 123–130.
11. Wei, X.; Lv, J.; Sun, J.; Dong, E. GCLO: Ground Constrained LiDAR Odometry with Low-drifts for GPS-denied Indoor Environments. In Proceedings of the 2022 IEEE International Conference on Robotics and Automation (ICRA), Philadelphia, PA, USA, 23–27 May 2022.
12. Coughlan, J.M.; Yuille, A.L. Manhattan World: Compass direction from a single image by Bayesian inference. In Proceedings of the Seventh IEEE International Conference on Computer Vision, Kerkyra, Greece, 20–27 September 1999; Volume 942, pp. 941–947.
13. Kim, P.; Coltin, B.; Kim, H. Visual Odometry with Drift-Free Rotation Estimation Using Indoor Scene Regularities. In Proceedings of the British Machine Vision Conference (BMVC), London, UK, 4–7 September 2017; pp. 62.1–62.12.
14. Li, Y.; Brasch, N.; Wang, Y.; Navab, N.; Tombari, F. Structure-SLAM: Low-Drift Monocular SLAM in Indoor Environments. *IEEE Robot. Autom. Lett.* **2020**, *5*, 6583–6590. [CrossRef]
15. Yunus, R.; Li, Y.; Tombari, F. Manhattanslam: Robust Planar Tracking and Mapping Leveraging Mixture of Manhattan Frames. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA), Xi'an, China, 30 May–5 June 2021.
16. Company-Corcoles, J.P.; Garcia-Fidalgo, E.; Ortiz, A. MSC-VO: Exploiting Manhattan and Structural Constraints for Visual Odometry. *IEEE Robot. Autom. Lett.* **2022**, *7*, 2803–2810. [CrossRef]
17. Zhou, H.; Zou, D.; Pei, L.; Ying, R.; Liu, P.; Yu, W. StructSLAM: Visual SLAM With Building Structure Lines. *IEEE Trans. Veh. Technol.* **2015**, *64*, 1364–1375. [CrossRef]
18. Zou, D.; Wu, Y.; Pei, L.; Ling, H.; Yu, W. StructVIO: Visual-inertial odometry with structural regularity of man-made environments. *IEEE Trans. Robot.* **2019**, *35*, 999–1013. [CrossRef]
19. Dai, A.; Lund, G.; Gao, G. PlaneSLAM: Plane-based LiDAR SLAM for Motion Planning in Structured 3D Environments. *arXiv* **2022**, arXiv:2209.08248.
20. Fischler, M.; Bolles, R. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun. ACM* **1980**, *24*, 381–395. [CrossRef]
21. Yang, M.Y.; Förstner, W. Plane detection in point cloud data. In Proceedings of the 2nd International Conference on Machine Control Guidance, Bonn, Germany, 9–11 March 2010; Volume 1, pp. 95–104.
22. Borrmann, D.; Elseberg, J.; Kai, L.; Nüchter, A. The 3D Hough Transform for plane detection in point clouds: A review and a new accumulator design. *3D Res.* **2011**, *2*, 3. [CrossRef]
23. Vo, A.V.; Truong-Hong, L.; Laefer, D.F.; Bertolotto, M. Octree-based region growing for point cloud segmentation. *ISPRS J. Photogramm. Remote Sens.* **2015**, *104*, 88–100. [CrossRef]
24. Taguchi, Y.; Jian, Y.D.; Ramalingam, S.; Chen, F. Point-plane SLAM for hand-held 3D sensors. In Proceedings of the 2013 IEEE International Conference on Robotics and Automation, Karlsruhe, Germany, 6–10 May 2013.
25. Wilmott, C.J.; Matsuura, K. Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. *Clim. Res.* **2005**, *30*, 79–82. [CrossRef]
26. Kaess, M.; Johannsson, H.; Roberts, R.; Ila, V.; Leonard, J.J.; Dellaert, F. iSAM2: Incremental smoothing and mapping using the bayes tree. *Int. J. Robot. Res.* **2011**, *31*, 216–235. [CrossRef]
27. Grupp, M. evo: Python Package for the Evaluation of Odometry and SLAM. 2017. Available online: https://github.com/MichaelGrupp/evo (accessed on 15 November 2022).