Ground Constrained 3D Lidar SLAM Design and Implementation

Wenwen Tian[1,3], Puwei Yang [1], Chao Yang [2], Sisi Zlatanova[3]

[1] School of Geography and Information Engineering, China University of Geosciences(Wuhan), Wuhan, China-tianww@cug.edu.cn -1202121142@cug.edu.cn
[2] National Engineering Research Center for Geographic Information System, China University of Geosciences(Wuhan), Wuhan, China-yangchao@cug.edu.cn
[3] School of Built Environment, University of New South Wales, Sydney, Australia-wenwen.tian, s.zlatanova@unsw.edu.au

Keywords: Synchronized Positioning, Multi-line LiDAR, Factor Graph, Ground Feature Extraction, SLAM.

Abstract

Synchronized positioning and mapping techniques based on multi-line LIDAR are mainly used for mobile platforms that require high-precision positioning, such as UAVs and self-driving cars. However, multi-line LiDAR is limited by vertical field of view and vertical angular resolution, which can affect the vertical accuracy of attitude estimation. In this paper, we aim to extract the ground information in the environment and fuse it into a factor map to constrain the vertical accuracy of the attitude. Different from the traditional geometric information-based ground plane segmentation, this paper utilizes point cloud distribution information to extract the ground cloud and uses high-confidence ground features to constrain the attitude drift by considering their distribution weights. Finally, the laser range factor, ground constraint factor and closed-loop factor are integrated under the factor graph optimization framework. The effectiveness of the method is validated on a self-constructed dataset.

1. Introduction

The mainstream 3D laser SLAM methods mostly adopt the idea of feature matching. Among them, LOAM (Lidar Odometry and Mapping) utilises a feature-based scanning matching approach using point cloud data collected by LiDAR for position estimation (Zhang and Singh, 2014). The LOAM method achieves coarse positioning by means of a high-frequency laser odometer and is calibrated using a low-frequency laser odometer. However, the LOAM method mainly focuses on the implementation of the odometer and does not include a back-end optimisation component. Deschau proposed the IMLS-SLAM (Iterative Closest Point with Maximum Likelihood Estimation for SLAM) method in 2018 (Deschaud, 2018), which utilises the constraint relationship between the sampling points and the implicitly moving least squares surface to achieve point cloud alignment. The bit position estimation method of IMLS-SLAM is similar to LOAM. Shan and Englot proposed a lightweight SLAM algorithm based on LOAM in 2018, LeGO-LOAM (Lightweight and ground-optimised LOAM) (Shan and Englot, 2018), which extracts the ground point cloud geometrically and constructs planar constraints based on the ground point cloud, and solves the six-degree-of-freedom position using Levenberg-Marquardt (LM) (Gavin, 2013) optimisation method with loopback detection and back-end optimisation module. Based on LeGO-LOAM, Shan conducted further research and proposed LIO-SAM in 2020(Shan et al., 2020), which adds IMU pre-integration factor and GPS factor in the back-end optimisation module, eliminates the accumulated error by using the respective characteristics of multi-sensors, and adopts an incremental global optimisation of the factor graph, which further improves the accuracy and robustness of the system. However, none of the above articles considered the vertical resolution defect problem of LiDAR itself.

Due to the sampling principle of the multi-line LIDAR, the fine grain of the data in the vertical direction is much smaller than that in the horizontal direction, which brings about the difference between the constraint information in the vertical and horizontal directions. In the process of moving the mobile platform, with the accumulation of errors, this difference in laser SLAM is manifested in the Z-axis drift phenomenon, i.e., the positioning of the mobile platform in the Z-axis direction has a large deviation, and the constructed environment map appears to be shifted upward or downward to a certain extent. As an important plane in the environment, the ground has more stable structural characteristics, and many laser points are located on the ground, so the ground constraints play a key role in the position estimation of the mobile platform, which can inhibit the Z-axis drift to a certain extent. The LeGO-LOAM algorithm adopts a two-step optimization method based on the ground features and edge features, and the six degrees of freedom of the mobile platform's inter-frame pose are split up: ground The ground features are used to constrain the Z, Roll and Pitch degrees of freedom in the pose estimation, and the edge features are used to constrain the X, Y and Yaw degrees of freedom in the pose estimation.

The constraints of the ground features help to mitigate the Z-axis drift phenomenon to obtain more accurate localization and environment maps, but due to its geometrically based ground segmentation, the ground point cloud cannot be extracted efficiently in the regions with large ground fluctuations. Based on LeGO-LOAM, the Groud-SLAM algorithm adopts the weighted least squares method to realize the ground segmentation, and uses the correspondence between the ground of the current frame and the ground of the local map as the constraints to reduce the pose errors in the three degrees of freedom of Z, Roll and Pitch. The HDL-Graph-SLAM algorithm utilizes ground constraints to correct the cumulative rotational error that is difficult to compensate for loopback detection, but it requires the existence of a uniform ground plane in a large indoor public environment and cannot be applied to complex scenes such as outdoors.

The main contributions of this paper are as follows:

1: A factor graph optimization framework containing ground constraints is proposed to constrain the position of an autonomous mobile platform and strengthen its Z-axis constraints.

2: Utilizing a self-built dataset containing LiDAR data and imu sequences, we validate our proposed algorithm in a real scenario.

2. Methodology

2.1 SLAM Front-End Part

2.1.1 Feature Extraction: Multi-line lidar (Yang et al., 2022) can collect tens of thousands of point data in one scan. The huge amount of data brings a huge computing load to the SLAM system, thus affecting the real-time performance of SLAM. Based on this, this chapter draws on the feature extraction method in LOAM, starting from the smoothness of the laser point, to construct edge features and plane features is a point on the radar scanning line beam. The smoothness of this point is shown in the following formula:

$$c = \frac{1}{|n| \cdot p_{(r,i)}} \left\| \sum_{j \in n, j \neq i} \left[p_{(r,i)} - p_{(r,j)} \right] \right\| \tag{1}$$

Where c is the smoothness of p_i, n represents the set of adjacent points in the scanning beam centred on p_i, p_j is another point in n except point p_i. When extracting geometric features from point clouds, it is necessary to remove unreliable feature points that are easily occluded or disappear. As shown in Figure 1, the plane of point B in Figure 1(a) is almost parallel to the incident light. The plane of point A in Figure 1(b) is easily blocked by the plane of B. In the process of extracting features, these two types of points are Cull.

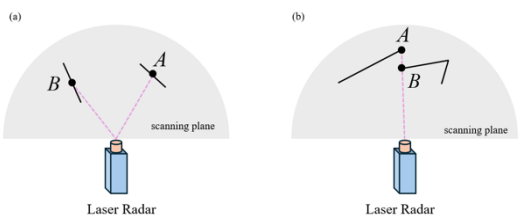


Figure 1. Unreliable point

The point cloud features are shown in Figure 2, where purple is the plane feature and green is the edge feature. Based on the above characteristics, this chapter selects the geometric features in the local map and the geometric features of the current frame to estimate the pose of the odometry. In order to improve the efficiency of feature matching, this section uses the KD-tree(Yang et al., 2022) data structure to divide the point cloud data and accelerate the matching process.

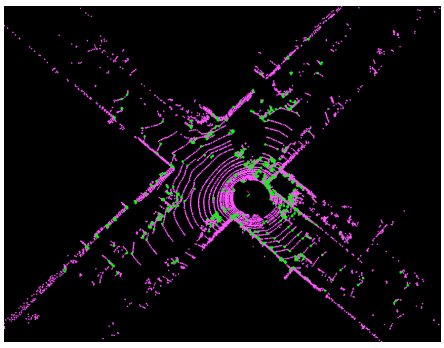


Figure 2. Point and line feature

2.1.2 Coarse Odometry: Based on the features extracted from the above content, the relative motion can be calculated from the edge features and surface features extracted from the two adjacent frames of data in a scan matching manner, and the pose of the mobile platform, that is, the lidar mileage, can be effectively evaluated from the two adjacent frames of data. count. Both the current frame and the local map contain planar features and edge features, and these two features are matched respectively. Denote the feature set in the current frame as S_t, where S_t^e and S_t^p respectively represent the edge feature and plane feature sets. Submap M_t consists of S_t, as shown in the following formula:

$$M_t = \left\{ M_t^e, M_t^p \right\} \tag{2}$$

$$M_t^e = S_t^{w,e} \bigcup S_{t-1}^{w,e} \bigcup S_{t-2}^{w,e} \bigcup \ldots \bigcup S_{t-n}^{w,e} \tag{3}$$

$$M_t^p = S_t^{w,p} \bigcup S_{t-1}^{w,p} \bigcup S_{t-2}^{w,p} \bigcup \ldots \bigcup S_{t-n}^{w,p} \tag{4}$$

Where M_t^e represent the set of edge features and M_t^p represent the set of plane features, $S_t^{w,e}$ is the edge feature of S_t^e in the global coordinate system after pose initialization, $S_t^{w,p}$ is the edge feature of S_t^p in the global coordinate system after pose initialization. In summary, based on the above characteristics, using ICP (Wei et al., 2020) in LIO-SAM to solve the pose of the current frame.

2.2 SLAM Back-End Part

Ground segmentation extracts ground plane information from raw point cloud data. For laser SLAM systems operating outdoors, the ground is a relatively stable plane, which can provide a more accurate Z-axis reference for the laser SLAM system, provide more accurate positioning and attitude estimation for mobile platforms operating in outdoor environments to ensure Z-axis data quality, reducing Z-axis drift.

Cloth simulation filtering is a filtering method that simulates the characteristics of cloth in the real world. For point cloud data, this method can consider the interactions and constraints between points in the point cloud, making the point cloud data smoother, continuous and more realistic. Compared with traditional ground segmentation methods (Zhang et al., 2016), cloth simulation filtering has the advantages of environmental universality, scalability, and high accuracy (Li et al., 2019). This section uses this method to extract ground point clouds in key frames and construct ground constraints.

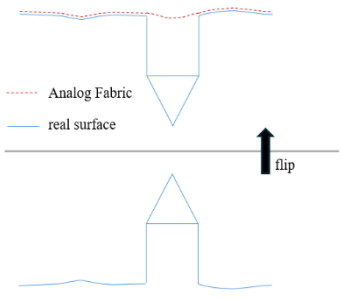


Figure 3. Schematic diagram of cloth simulation filtering

2.2.1 Ground constraints: After the cloth simulation filter flips the point cloud, the cloth falls freely and covers the point cloud. The ground point cloud is extracted by simulating the ground contour information of the point cloud. The cloth in the above algorithm is a grid model. Its construction method is the same as the cloth principle in the real world. Each particle with a fixed mass and negligible volume is connected by a connecting line with a certain stretch ability to form a particle spring model. The cloth particle spring model is shown in Figure 4:

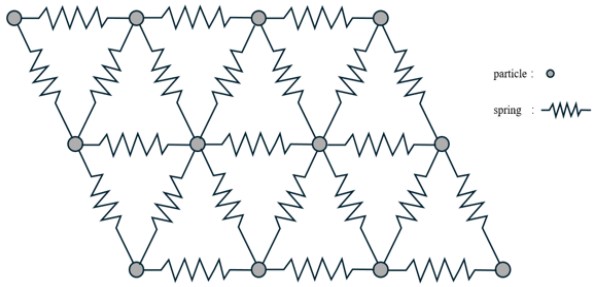


Figure 4. Cloth particle spring model

Cloth simulation filtering uses the cloth position after free fall as the ground model. For cloth particles, each cloth particle is only affected by gravity and the interaction force between cloth particles.

$$m\frac{\partial \mathrm{X}(t)}{\partial t^2} = F_{ext}\left(\mathrm{X},t\right) + F_{\mathrm{int}}\left(\mathrm{X},t\right) \tag{5}$$

(Liu et al., 2020)

Where X is the position of cloth particles at t time, $F_{ext}\left(\mathrm{X},t\right)$ represents the influence of external factors on the cloth protons, $F_{\mathrm{int}}\left(\mathrm{X},t\right)$ represents the influence of internal driving factors of cloth. According to the above formula, in order to obtain the accurate position of the cloth at a certain moment, it is necessary to analyze the motion model of the cloth particles. When considering the influence of gravity, according to Newton's second law, the motion model of the cloth particle is as follows:

$$\mathrm{X}\left(t+\Delta t\right) = 2\mathrm{X}\left(t\right) - \mathrm{X}\left(t-\Delta t\right) + \frac{G}{m}\Delta t^2 \tag{6}$$

Where $\mathrm{X}\left(t\right)$ is the position of cloth particles at t time, the time step of cloth particle motion is Δt, G is a constant, m represents the mass of the cloth particles. It can be seen from the above that when the initial position and movement duration of the cloth particle are known, the current position of the cloth particle can be calculated. If the position of the cloth particle reaches or crosses the laser point, the particle will be marked as an immovable particle. If the position of the cloth particle reaches or crosses the laser point, the particle will be marked as an immovable particle. On the basis of considering the gravity factors mentioned above, it is also necessary to consider the interaction between the cloth particles. Since there are irregular objects such as buildings, trees, and public facilities in the point cloud data, holes will be formed on the surface after the point cloud is flipped, interrupting the continuity of the data. In order to constrain the displacement of the cloth particles, the cloth particles need to be constrained by the force between the cloth particles.

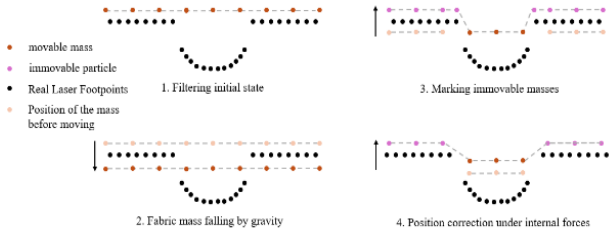


Figure 5. Movement of cloth particles

Figure 5 shows the schematic diagram of movement of cloth particles. When two adjacent cloth particles are both movable particles, the displacement of the two particles needs to be corrected in the opposite direction of gravity. If there is only one movable particle, the particle needs to move to the parallel position of the immovable particle. If two adjacent cloth particles have the same height, both particles are in a stable state and the particles do not move. During the operation of the algorithm, the displacement of the cloth particles needs to be repeatedly corrected according to this process. The displacement can be calculated by equation 7:

$$\vec{d} = \frac{1}{2}b\left(\overrightarrow{p_i} - \overrightarrow{p_0}\right)\cdot\vec{n} \tag{7}$$

Where $\vec{d}$ is the displacement of the cloth particle, b represents the movable attribute of the cloth particles. When b is 1, it means that the particle is a movable particle; when b is 0, it means that the particle is an immovable particle. $\overrightarrow{p_i}$ is the adjacent cloth particle of $\overrightarrow{p_0}$; $\vec{n}$ is the unit vector $\left(0,0,1\right)^T$ in the vertical direction. The strength of the interaction between cloth particles depends on the cloth hardness parameter. The harder the cloth, the stronger the interaction between the particles, and the greater the constraints between them. For example, assuming that the height difference between the movable particle and the immovable particle is d , and the cloth hardness is 1, 2 and 3 respectively, the displacement of the movable particle is as follows:

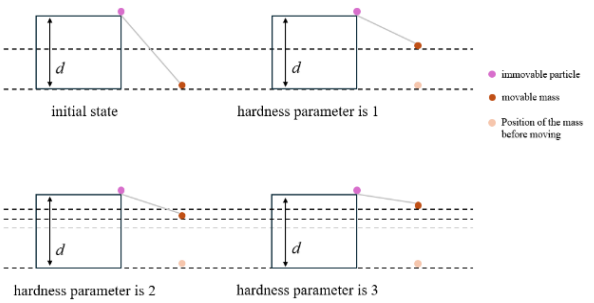


Figure 6. Effect of particle internal force under different fabric hardness parameters

2.2.2 IMU pre-integration: IMU has a high measurement frequency (above 100HZ) (Wu et al., 2021). If real-time IMU data is used for inertial calculation during the fusion process with lidar, the mobile platform needs to have strong computing capabilities, which is not conducive to the real-time operation of the SLAM system, therefore IMU pre-integration(Liu et al., 2020) is often used in SLAM to process IMU data, which can reduce the demand for real-time IMU data processing, reduce the amount of calculation and delay, and improve the consistency and stability of IMU data while meeting the requirements of system pose change detection, reducing the impact of IMU noise and error on pose. Based on the above advantages, IMU pre-integration has been widely used in SLAM. This section deduces the mathematical model of IMU pre-integration based on the original IMU data. Based on the above advantages, IMU pre-integration has been widely used in SLAM. This section deduces the mathematical model of IMU pre-integration based on the original IMU data. Assuming that the IMU data between time i and j is known, using Euler integration (Capobianco et al., 2017) to integrate the IMU measurement data, and the following formula can be obtained:

$$R_j = R_i \prod_{k=i}^{j-1} \left(Exp\left(\left(\tilde{\omega}_k - b_{g,k} - \eta_{gd,k} \right) \Delta t \right) \right) \tag{8}$$

$$\begin{aligned} v_j &= v_i + g\Delta t_{ij} \\ &+ \sum_{k=i}^{j-1} R_k \left(\tilde{a}_k - b_{a,k} - \eta_{ad,k} \right) \Delta t \end{aligned} \tag{9}$$

$$\begin{aligned} p_j &= p_i + \sum_{k=i}^{j-1} v_k \Delta t + \frac{1}{2} g \Delta t_{ij}^2 \\ &+ \frac{1}{2} \sum_{k=i}^{j-1} R_k \left(\tilde{a}_k - b_{a,k} - \eta_{ad,k} \right) \Delta t^2 \end{aligned} \tag{10}$$

Where Δt_{ij} is the time accumulation during the movement of the mobile platform, R_j , v_j and p_j respectively represent the rotation matrix, velocity and displacement at time j , R_i , v_i and p_i respectively represent the rotation matrix, velocity and displacement at time i .Based on the known IMU measurement data between the state at time i and j , the above formula can derive the motion state at time j . In the above integration process, the direct integration of IMU data can only optimize the current state. When the historical data changes, the data at all times need to be recalculated. In order to overcome this problem, the IMU pre-integration method is used to update the status at each moment in real time. Define the relative motion between i time and j time:

$$\Delta R_{ij} \triangleq R_i^T R_j = \prod_{k=i}^{j-1} Exp\left(\left(\tilde{\omega}_k - b_{g,k} - \eta_{gd,k} \right) \Delta t \right) \tag{11}$$

$$\begin{aligned} \Delta v_{ij} &\triangleq R_i^T \left(v_j - v_i - g\Delta t_{ij} \right) \\ &= \sum_{k=i}^{j-1} R_{ik} \left(\tilde{a}_k - b_{a,k} - \eta_{ad,k} \right) \Delta t \end{aligned} \tag{12}$$

$$\begin{aligned} \Delta p_{ij} &\triangleq R_i^T \left(p_j - p_i - v_i \Delta t_{ij} - \frac{1}{2} g \Delta t_{ij}^2 \right) \\ &= \sum_{k=i}^{j-1} \left[\Delta v_{ik} \Delta t + \frac{1}{2} \Delta R_{ik} \left(\tilde{a}_k - b_{a,k} - \eta_{ad,k} \right) \Delta t^2 \right] \end{aligned} \tag{13}$$

The above formula represents the entire pre-integration process.

2.2.3 Loop Closure Detection: In the process of real-time odometry calculation and incremental map construction of the SLAM system, the continuous accumulation of pose estimation errors causes the odometry and map to drift, and even causes the SLAM system to fail. In order to reduce the impact of error accumulation on positioning accuracy, this section constructs a loopback factor based on Euclidean distance to detect whether there is historical data during this operation near the mobile platform. If it exists, using the currently collected environmental data and historical data of the mobile platform to construct a constraint relationship to construct a loopback factor.
Based on this constraint, the pose of the current mobile platform is optimized to eliminate the impact of accumulated errors on pose estimation. The optimization results are shown in Figure 7:

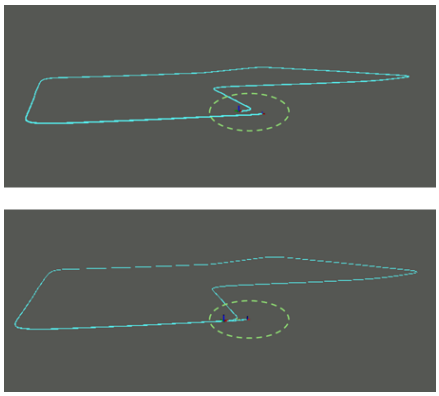


Figure 7. Loopback detection function

2.2.4 Factor Graph Optimization Incorporating Ground Constraints

This section introduces the factor graph optimization model based on the above constraints. In order to effectively integrate lidar odometry, IMU pre-integration, ground constraints and loop detection, we use the factor graph model to abstract sensor fusion into a nonlinear optimization problem(Lu and Milios, 1997). Traditional filtering-based sensor fusion methods, such as Kalman filter(Singh et al., 2024) and its nonlinear extensions of extended Kalman filter, unscented Kalman filter and other variants(Han-hai et al., 2017; Junior et al., 2022; Pan et al., 2024), have the problem of difficult data association in high-dimensional data space. The main advantage of factor graphs is re-linearization and iteration, which can bring the optimization state iteratively closer to the optimal state. The basic idea in factor graph optimization that incorporates ground constraints is to associate the system state (mobile platform state and key frames) with observation data (lidar scanning data and inertial measurement data) over a period of time and build a graph model. Global optimization is achieved based on Bayes' theorem(Li Shuai-Xin et al., 2021).

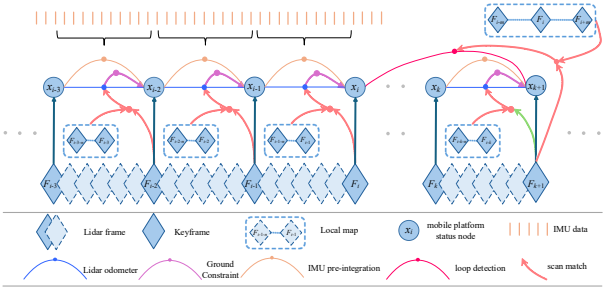


Figure 8. Algorithm running process

For system states and observations at any period of time, the maximum posterior probability of the joint distribution probability density of all random variables can be calculated by constructing a factor graph model, as follows:

$$\begin{aligned} X^{\mathrm{MAP}} &= \arg\max_{X}\left(p\left(X \mid Z\right)\right) \\ &= \arg\max_{X}\prod_{i}\phi_i\left(x_i\right) \end{aligned} \tag{14}$$

Where X^{MAP} represents the maximum posterior probability estimate of the state to be estimated, X represents the unknown state variable in the current factor graph model, p represents the joint probability distribution function in the current factor graph model, ϕ is a factor node, x is a variable node. This section integrates the factor graph optimization method with ground constraints, describes the SLAM problem as factor nodes and variable nodes in the factor graph model, and achieves global optimization based on factor graph inference.
X_k is defined as the state variable at time k :

$$X_k = \left[\left(R_B^W\right)^T, \left(p_B^W\right)^T, \left(v_B^W\right)^T, b_g^T, b_a^T\right]^T \tag{15}$$

In this chapter, the purpose of directly using the positioning information state variable instead of its error for optimization mainly includes the following aspects:

1: Positioning information state variables can directly describe the dynamic modelling process of the system through the model in Figure 8;
2: To reduce the impact of sensor noise, the errors in sensor measurements are modeled in error terms and passed to the factor graph. However, these error terms may be affected by sensor noise and introduce uncertainty. Using state variables directly can reduce the dependence on sensor noise, improving the accuracy of state estimation.
3: After pre-integrating the IMU measurement values, the pre-integration factors can be directly constructed and added to the factor graph optimization.

In order to ensure the stability of the algorithm, incremental smoothing rather than offline maximum likelihood method is used in this article. The purpose of this is to limit the odometry error in time during the real-time operation of the SLAM system and avoid falling into a local minimum in the subsequent optimization process. This article adopts the sliding window method for processing, which achieves real-time performance in the SLAM algorithm while ensuring the accuracy of pose estimation and map. As shown in Figure 8, two adjacent poses x_{i-1}, x_i and are connected by the laser inertial odometry factor p, the ground constraint factor , and the IMU pre-integration factor . The relationship between two adjacent posture nodes and their conditional probability model can be expressed as:

$$\begin{aligned} x_i = f\left(x_{i-1}, p_{DR-odom,i-1}, p_{ground,i-1},\right. \\ \left. p_{imuPre,i-1}\right) + \omega_i \end{aligned} \tag{16}$$

$$\begin{aligned} &P\left(x_i \mid x_{i-1}, p_{DR-odom,i-1}, p_{ground,i-1}, p_{imuPre,i-1}\right) \\ &= N\left(f\left(x_{i-1}, p_{DR-odom,i-1}, p_{ground,i-1},\right.\right. \\ &\left.\left. p_{imuPre,i-1}\right), \Omega_{i-1}\right) \end{aligned} \tag{17}$$

Where Ω_{i-1} represents zero-mean Gaussian(Yang et al., 2023) additive noise ω_i. The loopback factor indicates that the mobile platform revisits the constraints constructed by the historical location. When a loop is observed, the following formula expresses the relationship between the loop constraint $l_{i,k+1}$ and the mobile platform pose x_i , x_{k+1} . $l_{i,k+1}$ represent the pose change relationship between the current observation of the mobile platform at x_i and the corresponding submap of x_{k+1} .

$$x_{k+1} = g\left(x_i, l_{i,k+1}\right) + \beta_{i,k+1} \tag{18}$$

$$P\left(x_{k+1} \mid x_i, l_{i,k+1}\right) = N\left(g\left(x_i, l_{i,k+1}\right), \theta_{i,k+1}\right) \tag{19}$$

where $\theta_{i,k+1}$ is the covariance corresponding to Gaussian noise $\beta_{i,k+1}$ .Equation 20 integrates all mobile platform pose nodes and their corresponding constraints:

$$U = \left\{p_{DR-odom,i}, p_{ground,i}, p_{imuPre,i}, l_{i,k+1}\right\} \tag{20}$$

We can express the posterior probability within a certain sliding window as:

$$\begin{aligned} P\left(X \mid U\right) \propto \prod P\left(x_i \mid x_{i-1}, p_{DR-odom,i-1},\right. \\ \left. p_{ground,i-1}, p_{imuPre,i-1}\right) \\ \prod P\left(x_{k+1} \mid x_i, l_{i,k+1}\right) \end{aligned} \tag{21}$$

By maximizing the above representation, the optimization problem is incrementally solved using the iSAM2(Kaess et al., 2012) algorithm in the form of maximum a posteriori estimation:

$$\begin{aligned} X^{\mathrm{MAP}} &= \arg\max_{X}\left(P\left(X \mid U\right)\right) \\ &= \arg\min_{X}\left(-\log\left(P\left(X \mid U\right)\right)\right) \\ &= \arg\min_{X}\left(\sum_i \left\| f\left(x_{i-1}, p_{DR-odom,i-1}, p_{ground,i-1}, p_{imuPre,i-1}\right) - x_i \right\|_{\Omega_{i-1}}^2\right. \\ &\left. + \sum_i \left\| g\left(x_i, l_{i,k+1}\right) - x_{k+1} \right\|_{\theta_{ij}}^2\right) \end{aligned} \tag{22}$$

3. Experimental Results and Analysis

In order to verify its effectiveness in real scenarios, the team has built an experimental platform for positioning and mapping, using it to collect campus data to verify the effectiveness of the above algorithm.

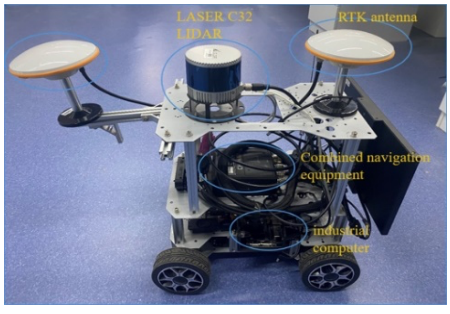


Figure 9. LIO-SAM Hardware Platform

3.1 Platform and Dataset

The hardware platform includes a 32-line lidar. This sensor has a horizontal field of view, a vertical field of view and a detection range of two hundred meters. It can effectively detect and identify objects in the surrounding environment and has all-round perception capabilities. The main specifications are shown in Table 1:

Item		Parameter
Wires		32
Theory		Time of Flight method
Scan range		200m
Scanning accuracy		±3cm
Scanning frequency		20Hz
field of view	Horizontal	360°
	Vertical	−16° ~ 15°
Angular resolution	Horizontal	0.36°
	Vertical	1°
Measurement（D*H）		Φ120*110mm
Weight		1.6kg

Table 1. The main parameters of LSLiDAR C32

The IMU and RTK data selected for the relevant experiments in this chapter come from POMS-GI420Pro, a device produced by ShangHai HuaCe Navigation Technology LTD. It is a combined navigation device that integrates inertial measurement data and satellite positioning data. This device can provide a variety of navigation parameters and can achieve centimetre-level positioning. With the built-in MEMS inertial measurement sensor, it can provide three-axis acceleration values and angular velocity values. The main specifications of POMS-GI420Pro are shown in the table 2:

Classification	Item	Parameter
System accuracy	Attitude accuracy	0.1°
	Positioning accuracy	RTK:1cm+1ppm
	Frequency	100Hz
Gyroscope indicators	Measuring range	±400° / s
	Bias stability	6° / h
Accelerometer indicators	Measuring range	±8g
	Bias stability	0.02mg

Table 2. The main parameters of POMS-GI420Pro

The industrial computer model selected for the experimental platform in this chapter is Jetson Xavier NX, an embedded system development kit produced by NVIDIA. It is equipped with a 6-core NVIDIA Carmel ARM CPU and a 384-core NVIDIA Volta GPU with 48 Tensor cores, and supports Linux systems. The machine is small and lightweight, has low power consumption and is easy to deploy.

In order to verify the effectiveness of the algorithm proposed in this article, this chapter uses a above self-built platform to collect data of sparse road sections with lake surface features, structured road sections such as libraries and teaching buildings, and unstructured road sections with dense grass and trees in the campus environment.

3.2 Localization Error Comparison

The algorithm proposed in this paper is named LIO-GC comparing it with LIO-SAM in the Self-built data set. We evaluated the performance of the proposed algorithm on the following datasets: Community, which encompasses complex community environments such as lakes, forests, and buildings; and Road, primarily assessing algorithm performance on driving roads, with its satellite map as follows:

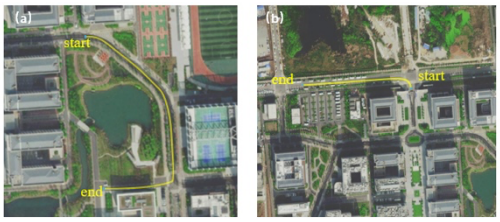


Figure 10. Test scenario with satellite view map

In this section, the validity of the front-end odometer of the LIO-GC algorithm proposed in this paper is verified by using the above data sets.

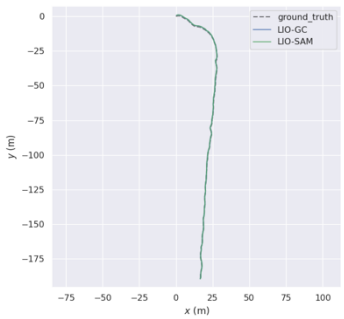


Figure 11. Keyframe trajectories of each algorithm

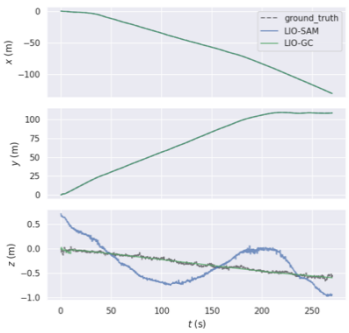


Figure 12. XYZ coordinate axis components of each algorithm

The figure above compares the positioning trajectories of the LIO-GC algorithm and the LIO-SAM algorithm on the Road dataset. It can be observed that in outdoor structured road driving scenarios, the LIO-SAM algorithm exhibits superior accuracy across various metrics. However, there is a certain degree of deviation from the ground truth on the Z-axis. In contrast, the proposed LIO-GC algorithm effectively enhances overall Z-axis positioning accuracy, aligning closely with the Z-axis ground truth.

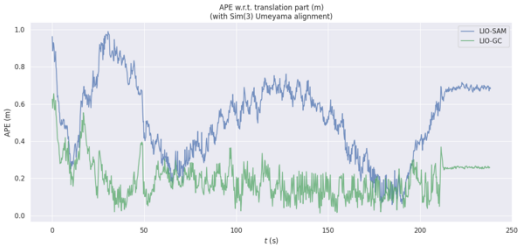


Figure 13. Absolute pose error of LIO-GC, LIO-SAM changes over time

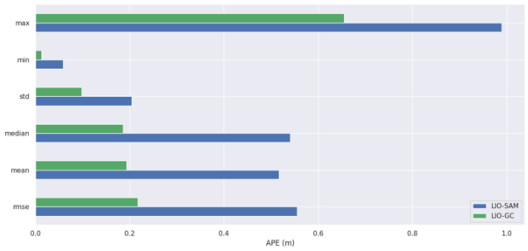


Figure 14. Mathematical statistics of absolute pose errors of LIO-GC and LIO-SAM

Figure 13 illustrates the variation of absolute pose error over time and the statistical analysis of absolute pose error for both the LIO-GC and LIO-SAM algorithms. As shown in the figure 13, during the initialization phase, the errors of the LIO-GC and LIO-SAM algorithms are relatively close. However, in the subsequent operation, the error of the LIO-GC algorithm quickly converges to a lower level and exhibits more stable fluctuations. Figure 14 presents the statistical analysis of absolute pose error for both algorithms. It is evident that the LIO-GC algorithm outperforms the LIO-SAM algorithm across multiple performance metrics.

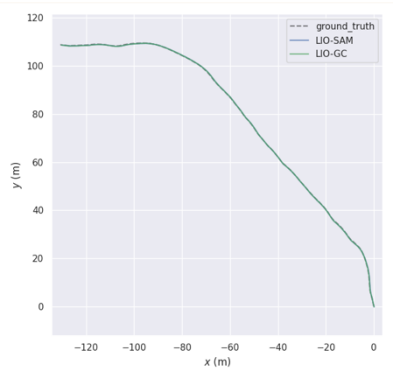


Figure 15. Keyframe trajectories of each algorithm

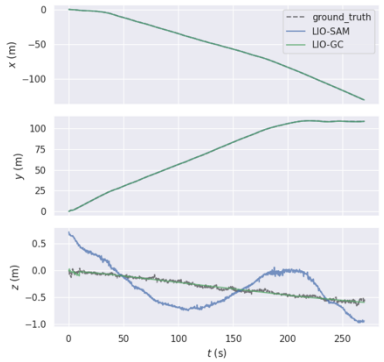


Figure 16. XYZ coordinate axis components of each algorithm

The above figure compares the positioning trajectories of the LIO-GC algorithm and the LIO-SAM algorithm on the community dataset. It can be observed that in complex driving scenarios involving lakes, forests, and buildings, the LIO-SAM algorithm exhibits more severe fluctuations on the Z-axis. This indicates a further degradation in the LIO-SAM algorithm's positioning capability in complex environments. In contrast, the LIO-GC algorithm demonstrates more stable performance. In Figure 17 below, consistent with the results on community dataset, the errors remain at a low level without significant fluctuations.

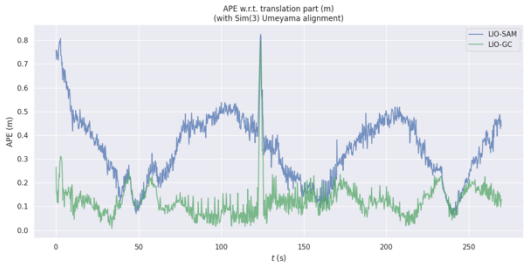


Figure 17. Absolute pose error of LIO-GC, LIO-SAM changes over time

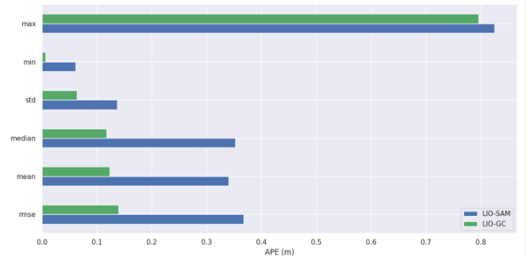
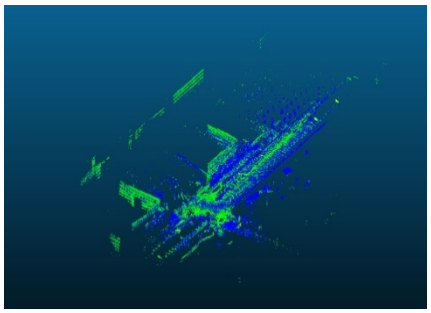
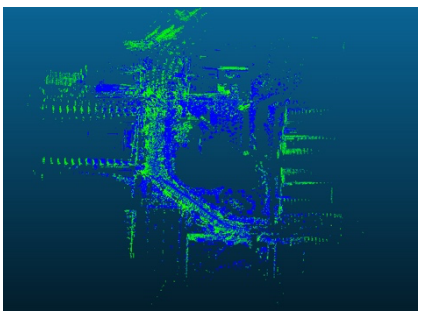


Figure 18. Mathematical statistics of absolute pose errors of LIO-GC and LIO-SAM

Figure 19 presents the global point cloud map reconstruction results of the LIO-GC algorithm on datasets road and community. From the figure, it is evident that the building contours are clearly visible, with distinct edges, indicating a clear and accurate mapping performance.



(a)Point cloud map on road data set



(b)Point cloud map on community data set

Figure 19. Point cloud map created by LIO-GC algorithm

As shown in Table 3, compared with the LIO-SAM algorithm, the LIO-GC algorithm proposed in this article has significantly improved in the RMSE, Mean, SSE and other indicators of absolute pose error. In the road data set, the LIO-GC algorithm has a better performance in RMSE The RMSE was reduced by 60.71%; in the community data set, the LIO-GC algorithm

reduced the RMSE by 63.16%. The above experiments show that LIO-GC can better compensate for the vertical resolution defect of lidar, effectively improve the positioning accuracy of mobile platforms, and better meet the positioning needs of mobile platforms in various environments.

Dataset	Algorithm	Rmse (m)	Mean(m)	SSE(m2)
road	LIO-SAM	0.56	0.52	351.6
	LIO-GC	**0.22**	**0.20**	**44.14**
community	LIO-SAM	0.38	0.34	176.98
	LIO-GC	**0.14**	**0.12**	**17.46**

Table 3. Error results of LIO-SAM and LIO-GC algorithms

4. Conclusions

In this paper, we introduced a Ground Constrained 3D LiDAR SLAM (LIO-GC) algorithm designed to address the vertical resolution limitations faced by mobile platforms requiring high-precision positioning, such as UAVs and autonomous vehicles. By extracting ground information from the environment and integrating it into a factor graph to constrain the vertical accuracy of the attitude, our method has demonstrated effectiveness on a self-constructed dataset. However, in unstructured scenes where the ground cannot be effectively extracted, the confidence of the ground factor is significantly reduced, potentially introducing new errors into the SLAM system.

However, in unstructured scenes where the ground cannot be effectively extracted, the confidence of the ground factor will be drastically reduced, so that new errors are introduced in the SLAM system. In subsequent research, the ground can be divided into grids, and the confidence of the ground point cloud of each grid can be evaluated. Only high-confidence ground data can be used, or the weight of the ground factors in factor graph optimization can be constructed based on this confidence to reduce the impact of this error on SLAM accuracy. Additionally, we will explore ways to further enhance the robustness of the algorithm in complex environments, such as by improving ground segmentation algorithms or incorporating other types of sensor data to assist in positioning.

References

Capobianco, G., Enea, M.R., Ferraro, G., 2017. Geometry and analysis in Euler’s integral calculus. *Arch. Hist. Exact Sci*. 71, 1–38.

Deschaud, J.-E., 2018. IMLS-SLAM: scan-to-model matching based on 3D data. *In 2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2480-2485.

Gavin, H.P., 2013. The Levenberg-Marquardt method for nonlinear least squares curve-fitting problems. Duke University.

Han-hai, S., Fa-jiang, H., Shu-wen, D., Kang-le, W., 2017. Research on slam algorithm of iterated extended Kalman filtering for multi-sensor fusion, *In Proceedings of the 3rd International Conference on Communication and Information Processing*, ACM, Tokyo Japan, pp. 242–246.

Junior, G.P.C., Rezende, A.M.C., Miranda, V.R.F., Fernandes, R., Azpurua, H., Neto, A.A., Pessin, G., Freitas, G.M., 2022. EKF-LOAM: An Adaptive Fusion of LiDAR SLAM With Wheel Odometry and Inertial Data for Confined Spaces With Few Geometric Features. *IEEE Trans. Automat. Sci,* 19, 1458–1471.

Liu, Z., Li, Y., Chen, C., 2020. Application of IMU Pre-integration in Variable-Height Lidar Odometry, *In 2020 4th International Conference on Robotics and Automation Sciences (ICRAS)*, Wuhan, China, pp, 112–116.

Li, Y., Li, C., Wang, S., and Fang, J., 2019. A CSF-modified filtering method based on topography feature. *Remote Sens. Technol. Appl*., 34, 1261–1268.

Lu, F., Milios, E., 1997. Globally Consistent Range Scan Alignment for Environment Mapping. *Autonomous Robots*, 4, 333–349.

Pan, H., Liu, D., Ren, J., Huang, T., Yang, H., 2024. LiDAR-IMU Tightly-Coupled SLAM Method Based on IEKF and Loop Closure Detection. *IEEE J. Sel. Top. Appl. Earth Observations Remote Sensing*, 17, 6986–7001.

Shuai-Xin, L., L. Guang-Yun, W. Li, Y. Xiao-Tian, 2021. LiDAR/IMU tightly coupled real-time localization method, *Acta Automatica Sinica*, 47(6), 1377-1389

Shan, T., Englot, B., 2018. LeGO-LOAM: Lightweight and Ground-Optimized Lidar Odometry and Mapping on Variable Terrain, *In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Madrid, pp. 4758–4765.

Shan, T., Englot, B., Meyers, D., Wang, W., Ratti, C., Rus, D., 2020. LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping, *In 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS),* IEEE, Las Vegas, NV, USA, pp. 5135–5142.

Singh, H., Mishra, K.V., Chattopadhyay, A., 2024. Inverse Unscented Kalman Filter. *IEEE Transactions on Signal Processing*, 72, pp. 2692-2709,

Wei, W., Shirinzadeh, B., Ghafarian, M., Esakkiappan, S., Shen, T., 2020. Hector SLAM with ICP Trajectory Matching, *In 2020 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*. IEEE, Boston, MA, USA, pp. 1971–1976.

Wu, Y., Niu, X., Kuang, J., 2021. A Comparison of Three Measurement Models for the Wheel-mounted MEMS IMU-based Dead Reckoning System. *IEEE Transactions on Vehicular Technology*, 70(11), 11193-11203

Yang, L., Sang, J., Sun, W., Liao, X., Sun, M., 2023. Nonstationary SBL for SAR Imagery Under Nonzero Mean Additive Interference. *IEEE Geosci. Remote Sensing Lett*. 20, 1–5.

Yang, T., Li, Y., Zhao, C., Yao, D., Chen, G., Sun, L., Krajnik, T., Yan, Z., 2022. 3D ToF LiDAR in Mobile Robotics: A Review. arXiv preprint arXiv:2202.11025.

Zhang, J., Singh, S., 2014. LOAM: Lidar Odometry and Mapping in Real-time, *In Robotics: Science and Systems*, 9, pp 1-5.

Zhang, W., Qi, J., Wan, P., Wang, H., Xie, D., Wang, X., Yan, G., 2016. An Easy-to-Use Airborne LiDAR Data Filtering Method Based on Cloth Simulation. *Remote Sensing,* 8, 501.