Mathematical Biosciences and Engineering

MBE, 20(5): 8954–8974.
DOI: 10.3934/mbe.2023393
Received: 26 November 2022
Revised: 11 February 2023
Accepted: 28 February 2023
Published: 09 March 2023

http://www.aimspress.com/journal/MBE

Research article

Research on simultaneous localization and mapping algorithm based on Lidar and IMU

Minghe Liu, Ye Tao*, Zhongbo Wang, Wenhua Cui and Tianwei Shi

School of Computer Science and Software Engineering, University of Science and Technology Liaoning, Anshan 114051, China

* **Correspondence:** Email: taibeijack@163.com; Tel: +8613304224928; Fax: +8604125929818.

Abstract: In recent years, the research of autonomous driving and mobile robot technology is a hot research direction. The ability of simultaneous positioning and mapping is an important prerequisite for unmanned systems. Lidar is widely used as the main sensor in SLAM (simultaneous localization and mapping) technology because of its high precision and all-weather operation. The combination of Lidar and IMU (Inertial Measurement Unit) is an effective method to improve overall accuracy. In this paper, multi-line Lidar is used as the main data acquisition sensor, and the data provided by IMU is integrated to study robot positioning and environment modeling. On the one hand, this paper proposes an optimization method of tight coupling of lidar and IMU using factor mapping to optimize the mapping effect. Use the sliding window to limit the number of frames optimized in the factor graph. The edge method is used to ensure that the optimization accuracy is not reduced. The results show that the point plane matching mapping method based on factor graph optimization has a better mapping effect and smaller error. After using sliding window optimization, the speed is improved, which is an important basis for the realization of unmanned systems. On the other hand, on the basis of improving the method of optimizing the mapping using factor mapping, the scanning context loopback detection method is integrated to improve the mapping accuracy. Experiments show that the mapping accuracy is improved and the matching speed between two frames is reduced under loopback mapping. However, it does not affect real-time positioning and mapping, and can meet the requirements of real-time positioning and mapping in practical applications.

Keywords: simultaneous positioning and mapping; multi-line lidar; factor graph; loopback detection

1. Introduction

1.1. Research background and significance

In recent years, with the rapid development of artificial intelligence technology, the Internet of Things big data and other high-tech achievements have been made. The progress of science and technology has improved people's quality of life. Robots begin to appear in more and more families. This has also become one of the symbols of the development of robot technology. As an important field of artificial intelligence, intelligent robot technology has attracted more and more scholars and researchers. The important embodiment of artificial intelligence technology is that the intelligence level and adaptability of robots have been greatly improved. Its applications have covered military, industrial, rescue, medical, etc. At present, large enterprises and scientific research institutions around the world regard the research of high IQ robot technology as a deeper frontier research. It aims to push the research of intelligent robots to a new height [1].

At the same time, in various fields, mobile robots are of great significance in people's work and life. People hope that mobile robots can play a greater role and have more functions. The working environment of mobile robots is becoming increasingly complex. Including underwater, tunnel, air and other operating environments. There are high requirements for the stability of mobile robot positioning and mapping in these complex environments, as well as adaptability to various environments. However, the existing work does not have a high-precision simultaneous location and algorithm research that is generally applicable to an outdoor environment, and the existing algorithms need to improve the accuracy of simultaneous location and mapping in an outdoor environment.

Location and mapping methods play an important role in the autonomous navigation of robots. Simultaneous positioning and mapping based on lidar are the main ways for robots to achieve autonomous navigation. Therefore, while building maps for a wide range of outdoor environments, the research on positioning and mapping methods with lidar as the main sensor has considerable practical and theoretical value.

In this paper, robot localization and mapping in unfamiliar environments are studied. On the basis of the existing point line matching algorithm and IMU data processing method, the localization and mapping algorithm of robots in an outdoor unfamiliar environment is studied. Due to various human interference and environmental emergencies. It is easy for a seemingly well-trained mobile robot to become at a loss. Therefore, a more stable and accurate robot positioning and mapping method is needed. In this paper, a method using a factor graph to fuse tightly coupled lidar and IMU is proposed. The lidar odometer generated by the front-end registration method based on point line feature extraction and the map created are optimized. On the basis of this factor graph, the concept of windowing is introduced to reduce the amount of calculation and improve the optimization speed. The edge probability method is used to retain the constraint relationship of the old frames removed in the sliding window. Finally, the Scan Context loopback detection method is combined with the back-end optimization method to reduce the mapping error. Then, the constraint relationship provided by loopback detection is put into the factor graph as a factor to strengthen optimization.

1.2. Research status of laser SLAM method

The SLAM problem originated in 1986 and is the beginning of the research on the combination of artificial intelligence and robotics. In the 1990s, the SLAM algorithm began to use filters, which is

a classic pose estimation algorithm and the mainstream algorithm at that time [2]. Smith et al. [3] proposed a SLAM algorithm based on the Extended Kalman Filter (EKF), whose application depends on the Gaussian noise hypothesis of the motion model and the observation model.

Doucet et al. [4] proposed the RBPF (Rao-Blackwellization Physical Filter) algorithm, which integrates the Rao-Blackwellization algorithm with particle filter and applies it to the SLAM field. Montemerlo et al. proposed the FastSLAM algorithm based on RBPF and EKF algorithm [5], which split the SLAM problem into two problems to solve. One problem is that the problem of robot localization is solved by the particle filter algorithm, and the other is that the problem of location estimation of environmental features is solved by EKF algorithm. In the implementation of the algorithm, a large number of particles will be sampled to fit the target distribution. Frequent resampling steps will lead to the reduction of particle diversity and waste a lot of computing resources. In order to solve the above problems, the FastSLAM 2.0 algorithm is proposed, and the Gaussian distribution assumption of particles is introduced into the FastSLAM algorithm, which effectively improves the efficiency of the algorithm and solves the problem of particle diversity reduction in RBPF algorithm [6]. It is one of the most efficient methods based on the filtering method in the current SLAM algorithm.

Grisetti et al. [7] proposed a GMapping algorithm to improve the proposed distribution and selective resampling based on RBPF. The improved proposed distribution of GMapping refers to the odometer information and the latest observation information, which can make the proposed distribution more accurate and closer to the target distribution. Set the threshold for selective resampling. When resampling, it is necessary to meet the set condition that the particle weight change exceeds the threshold. Use this method to reduce the number of resampling and improve the calculation efficiency. However, this algorithm is not used in large scene maps, because each particle carries a map, and the number of particles in the algorithm will increase with the increase of the scene, and the memory occupied and the amount of computation required to complete the algorithm will increase. This algorithm involves the balance of time complexity and space complexity. It is not suitable for building large scene maps by sacrificing space complexity to ensure time complexity.

Zhang and Singh [8] proposed the classic laser matching SLAM method, LOAM (Lidar Odometry and Mapping). This algorithm only needs 3D (Three Dimensions) laser point cloud data to achieve low drift and low computational complexity. The algorithm is divided into two parts. One is to use feature points to match data between adjacent point cloud frames under high frequency and low accuracy; The other part is the stitching of the point cloud data and the map after the correction based on the characteristic points are used by the lidar under the low-frequency operation. Its advantage is that it can separate the odometer and map building, use high frequency and low precision when processing each frame of data, and use low frequency and high precision when the number of frames is accumulated for a certain number of times, to ensure that the map can be drawn in real-time. The disadvantage is the lack of loopback detection.

The Google team proposed a set of SLAM algorithm Cartographer [9] methods based on graph optimization, the application range of which includes 2D and 3D laser radars. Cartographer uses odometer and IMU data to estimate the travel path of the car, gives the estimated value of the change of the car's position and attitude between successive frames, uses the estimated value to match the radar data, and further updates the estimated value of the car's position and attitude. A frame of radar data is superposed to form a sub-image after motion filtering, carry out global loopback detection and form a complete map of subgraphs after back-end optimization. The advantage is that the cumulative error is low, and the covariance matrix can be output naturally, but its memory consumption is large.

With the process of drawing, the amount of calculation continues to increase, and the computer requirements are high.

Shan and Englot [10] proposed an improved LEGO-LOAM algorithm based on LOAM. The difference between this algorithm and LOAM is that it classifies point clouds and divides a frame of point cloud data into point planes and line segments. First, a frame of point cloud data is projected onto the image. According to the angle of the laser radar, the image is evaluated column by column to find the ground point. The ground point does not participate in the subsequent classification. The point cloud matching optimization part is divided into two steps. First, the displacement, pitch angle and roll angle along the Z-axis can be obtained by using the plane feature optimization. The amount of plane feature optimization is used as the fixed constraint for the next step of optimization. The displacement and heading angle along the X and Y axes can be obtained by using the line feature optimization. The same accuracy can be obtained by the two steps of optimization, and the calculation time can be reduced by about 35%. Through these improvements, the algorithm can be used in various complex physical environments, and the computation required by the algorithm is particularly small. Droeschel and Behnke [11] proposed a point cloud matching method based on bin features, which is a continuous 3D laser simultaneous positioning and mapping algorithm. Compared with the traditional point-to-point ICP (Iterative Closest Point) algorithm, the bin feature can better represent the mean and variance of point cloud distribution in a certain region in space.

Behley and Stachniss [12] proposed a point cloud matching method SUMA (Surfel-based Mapping) algorithm based on bin features, which uses 3D laser scanning to integrate semantic information to promote the mapping process, but does not consider the point cloud distortion problem. Li et al. [13] proposed an algorithm for point cloud matching using NDT (Normal Distribution Transform). This algorithm aims at the problem of simultaneous location and mapping in the mine cave scene, without considering the impact of point cloud distortion. Deschaud [14] proposed a laser mileage calculation method called IMLS-SLAM, which uses the linear difference method to estimate the motion information at each moment and project it all to the first moment to eliminate the motion distortion. In addition, before point cloud matching, the author also removed the point cloud of the dynamic object, but the algorithm has low computational efficiency, and the effect of the image is far worse and flat in the environment of intense motion.

Shao et al. [15] proposed a SLAM method that integrates computer vision, laser radar and IMU [16]. This method adds cameras to solve the problem that the effect of using laser radar is not good in some degraded scenarios, and it still has good performance in non-degraded environments. Niu and Wang [17] proposed a multi-constraint factor graph optimization method for the unmanned vehicle application scenario. By taking the closed loop of ground, GPS (Global Positioning System) and point cloud features as constraint factors, the cumulative error generated by the laser odometer with the movement of the unmanned vehicle was eliminated. The algorithm used the KITTI data set to test whether the mapping effect was good [15]. Niu and Wang [17] proposed a multi-sensor SLAM framework, which is mainly aimed at automatic driving and applied to automatic driving vehicles. T. Shan et al. [18] proposed a close-coupled laser-inertial odometer method, which integrates the lidar odometer, IMU and GPS into a method of simultaneous positioning and mapping. This method has an excellent effect of simultaneous positioning and mapping outdoors, but uses a variety of sensors, and the equipment cost is high.

However, the existing work does not have a high-precision simultaneous location and algorithm research that is generally applicable to an outdoor environment, and the existing algorithms need to improve the accuracy of simultaneous location and mapping in an outdoor environment.

2. Relevant theories and methods

2.1. Laser radar TOF ranging principle

The TOF principle is to emit a laser beam through a laser transmitter. When the laser beam is irradiated on the object, it will reflect the light signal. Special sensors are used for reception. The time difference from launch to reception is measured directly. Multiply the speed of light to get the relative distance between the object and the object [19].

2.2. Laser radar motion dedistortion method

2.2.1. Laser radar self-motion distortion

Although lidar can quickly transmit and receive laser beams, each point forming the point cloud is not generated at the same time. Generally, the accumulated data within 100ms (corresponding to the typical value of 10 Hz) is output as a frame point cloud. If the absolute position of the laser radar body or the body where it is installed changes within this 100ms. Then the coordinate system of each point in this frame point cloud is different. Intuitively, the point cloud data of this frame will undergo certain "deformation". It cannot correspond to the detected environment information. Similar to shaking hands when taking photos, the photos will be burnt. This is the self-motion distortion of laser radar.

2.2.2. A method of removing the motion distortion of Lidar

The distortion of the point cloud is due to the fact that the radar carrier is not stationary when collecting a frame of point cloud data. It is the data collected in a moving state. Therefore, it is necessary to calculate the movement of the radar in the process of point cloud data acquisition. By compensating the relative time of each point in the point cloud data of each frame, the points in the point cloud of this frame are guaranteed to be in the same coordinate system. Compensation includes compensation of radar rotation and translation. Point cloud distortion methods include pure estimation method, sensor-aided method and fusion method [20–22].

The auxiliary method of using an IMU sensor, provides the carrier's speed, acceleration and other information through IMU. In the low-speed motion scene, the carrier can be assumed as a uniform motion model. That is, coordinate = motion × Time. The IMU information can also be used to correct the non-uniform part of the error in the uniform model assumption in high-speed moving scenes [23].

The time for lidar to collect a frame of point cloud data is known. By calculating the distance from each point in a frame point cloud to the lidar, the acquisition time of each point is obtained. The acquisition time of all points in a frame of the point cloud is unified under the scanning time of the first point of the point cloud. It is equivalent to unifying the acquisition position of all points at the initial position when the laser radar scans a frame.

The specific steps are. first, synchronize the LIDAR and IMU. Obtain the IMU data closest to the point cloud timestamp of a frame according to the time stamp of the lidar and IMU. Subtract time

stamps to calculate travel value. If the time difference is less than the synchronization threshold, it will be output as synchronization data. If the time difference is greater than the synchronization threshold, the data will be discarded, and the next frame of lidar sampling data will be taken to cycle the above process. Then the unordered point cloud is converted in order. Taking VLP-16 as an example, one frame of VLP-16 data is output in the form of the point cloud.

Specifically, each point in a frame point cloud has XYZ information. There is no other association between points. Therefore, the corresponding horizontal angle of the Scan corresponding to each point is unknown. Therefore, single-frame point cloud data is processed. Divide into 16 bundles. Record the harness corresponding to each point and the scanning time relative to the first point of the frame in the point cloud of this frame. The acceleration and Euler angle relative to the world coordinate system are obtained through a three-axis accelerometer and three-axis gyroscope in the IMU coordinate system. After removing the influence of gravity, the data obtained is the acceleration of IMU in the world coordinate system. Calculate the displacement and velocity of IMU data in each frame. Finally, the displacement and velocity of IMU data obtained are processed by the interpolation method. Calculate the compensation transformation matrix of each point in a frame point cloud relative to the starting point in the frame point cloud. The laser points of a frame of the point cloud are corrected by the compensation transformation matrix. The correction formula is the compensation transformation matrix multiplied by the laser point coordinates of each point in the frame scanning [24–26].

2.3. IMU model

IMU is a device for measuring the three-axis attitude Angle and acceleration of an object. It is composed of three-axis accelerometer and three-axis gyroscope, which can obtain high-frequency inertial measurement values [27].

3. Algorithm design

3.1. Optimization algorithm design of factor graph

3.1.1. Design of IMU factor diagram for lidar fusion

Lidar is usually used together with other sensors for state estimation and mapping. This design scheme can be divided into two categories according to the sensor fusion mode. Loose coupling and tight coupling. Compared with loosely coupled tightly coupled systems, which usually provide higher accuracy, it is currently the main focus of ongoing research. Therefore, this paper adds the IMU factor to the factor graph designed above for effect optimization. This can reduce the impact of radar odometer failure to a certain extent.

Three-dimensional lidar and a 6-DOF IMU are installed in the system. Use $R_c$and $P_c$to represent the relative rotation and translation between the two sensors respectively.R_c and P_c are used to project the lidar point to the corresponding IMU frame. As shown in the following (5).

The ranging equipment moves in the environment, provides N 3d points $x_i$at time $t_i(i = 1,\ldots,N)$, and divides them into M scans. Inertial data include a three-axis accelerometer and a three-axis gyroscope, which provide the original reading acceleration f_q and angular velocity w_q at time $t_q(q = 1,\ldots,Q)$. Based on the first lidar scan, a map consisting of P planes is considered fixed. After the first scan, each lidar point is associated with one of these planes. The plane associated with x is characterized by its normal vector n_i and its distance from the origin w_i. For n_i and w_i, there are

only P possible different values. In other words, if x_i and x_j belong to the same mapping plane, $n_i = n_j$ and $w_i = w_j$. In order to associate a single lidar point with the IMU reading, this paper uses GP return to independently infer the inertial reading of each IMU degree of freedom at any time t, $\hat{f}(t)$ and $\hat{\omega}(t)$.

The objective of this method is to estimate the calibration parameters R_c and P_c, IMU deviation, IMU direction R_W^m, position P_W^m and speed V_W^m, and the time shift δt between the two sensors. The subscript W represents the fixed world frame of the earth. The superscript m represents the m^{th}-scan from the lidar, and τ_m corresponds to the timestamp of the first lidar point in the m^{th} scan. Where represents the state to be estimated, deviation from $\hat{b}(f)$, $\hat{b}(\omega)$ and δt and time shift correction. P_W^0 is not part of the state, because you need to set an IMU position arbitrarily to define the world frame. Where $\hat{b}(f)$, $\hat{b}(\omega)$ are respectively the deviation d_i of $S = \{R_C, P_C, R_W^0, \ldots, R_W^M, P_W^1, \ldots, P_W^M, V_W^0, \ldots, V_W^M, \hat{b}_f, \hat{b}_\omega, \hat{\delta}t\}$ accelerometer and gyroscope, and δt is the time shift between the lidar and IMU data.

The calibration problem can be expressed as maximum likelihood estimation (MLE).

$$S^* = \underset{s}{argmin} -\log(p(SZ)) = \underset{s}{argmin}\, F(S) \tag{1}$$

Z is the available measurement and F is the optimization cost function. Under the assumption of zero mean Gaussian noise, it can be solved by minimizing the point plane distance corresponding to the lidar factor and the residual r_I^m corresponding to the IMU factor.

$$F(\mathrm{S}) = \sum_{\mathrm{m}=1}^{\mathrm{M}} r_I^{m2} \sum_r m + \sum_{i=1}^{N} d_i{}^2 \sum d_i \tag{2}$$

In Figure 1, node I_m represents the position and velocity of IMU at time t_m Node C is the calibration parameter set. The lidar factor expressed in square takes into account the distance from the reprojected point to the target plane. The black circle is the IMU pre-integration factor. The following symbols are used in the rest of this document, as shown in Formula (3).

$$\Delta t_k = t_{k+1} - t_k, \Delta\tau_m = \tau_{m+1} - \tau_m, \Delta\zeta_m^i = \tau_i - \tau_m \tag{3}$$

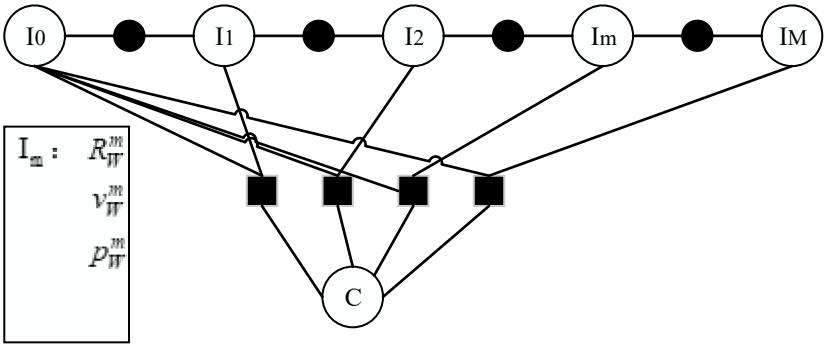


Figure 1. Lidar factor and IMU fusion factor diagram.

IMU pre integration measurement value is

$$\Delta p_m^i = \sum_{k=K}^{i-1} \Delta v_m^k \Delta t_k + \Delta R_m^k (f(t_k - \delta t) - b_f) \Delta t_k^2 \tag{4}$$

$$\Delta v_m^i = \sum_{k=K}^{i-1} \Delta R_m^k (f(t_k - \delta t) - b_f) \Delta t_k \tag{5}$$

$$\Delta R_m^i = \pi_{k=K}^{i-1} Exp((\omega(t_k - \delta t) - b_\omega) \Delta t_k) \tag{6}$$

When $\{K \in N | t_k = \tau_m\}$

$$p_m^i = p_W^m + \Delta \zeta_m^i v_W^m + \frac{1}{2} \Delta {\zeta_m^i}^2 g + R_W^m \Delta p_m^i \tag{7}$$

$$v_m^i = v_W^m + \Delta \zeta_m^i g + R_W^m \Delta v_m^i \tag{8}$$

$$R_W^i = R_W^m \Delta R_m^i \tag{9}$$

Radar factor, the projection error between two-point clouds of the lidar is defined as the distance from the point to the plane, which is used as the lidar factor. Represented as point x_i in the lidar frame F_L^i, the measured values need to be re projected into the first lidar frame F_L^0 according to the pre integration with IMU.

First, $\mathrm{x_i}$ is projected into IMU frame $\mathrm{F_L^i}$.

$$x_I^i = R_c x_i + p_c \tag{10}$$

Then $\mathrm{x_i^I}$ is projected into the world frame W.

$$x_W^i = R_W^i x_i^I + p_W^i \tag{11}$$

When $\tau_\mathrm{m} \leq \mathrm{t_i} < \tau_{\mathrm{m+1}}$,

$$x_{L_0}^i = {R_c}^T ({R_W^0}^T (x_i^w - p_w^0) - p_c) \tag{12}$$

Use $\mathrm{F_L^0}$ to indicate the distance from the point to the plane.

$$d_i = n_i^T x_{L_0}^i + w_i \tag{13}$$

where $\mathrm{i} = 1, \ldots, \mathrm{N}$ is the residual.

IMU factor, IMU factor can be seen as the constraint on IMU pose and speed. Relevant residuals can be directly obtained by manipulating (14) and (15).

$$r_p^m = R_w^{mT} (p_W^{m+1} - p_w^m - \Delta\tau_m v_w^m - \frac{{\Delta\tau_m}^2}{2} g) - \Delta p_m^{m+1} \tag{14}$$

$$r_R^m = Log({\Delta R_m^{m+1}}^T {R_w^m}^T R_W^{m+1}) \tag{15}$$

$$r_I^m = \left[r_R^m; r_v^m; r_p^m\right] \tag{16}$$

3.1.2. Design of sliding window optimization algorithm

When using the factor diagram to optimize the lidar odometer and map, but with the movement of the lidar, new point cloud data and IMU data continue to be introduced into the system, and when the location optimization or the number of lidar feature points become more, the calculation of the optimization process will be more complementary, and the optimization variables cannot be increased unlimited, but the use of sliding window technology, It is used to limit the large amount of calculation generated by a certain range of calculation methods.

VINS Mono, an algorithm of visual SLAM, applies a sliding window method. The algorithm first divides a video into keyframes and non-keyframes. The BA (Bundle Adjustment) problem of constructing images and signal points in keyframe images. Non keyframes are only used to estimate the pose transformation of the camera, that is, positioning. Not involved in drawing construction. Although not all video frames are applied, the calculation amount can be reduced. However, with the movement of the camera, the number of captured image data, keyframes, and the size of the map continue to increase. The calculation efficiency of batch optimization methods, such as BA, has been declining. In order to prevent this, some methods should be used to reduce or fix the size of BA. The method can be theoretical or engineering. For example, the simplest logic to control the size of BA is to keep the keyframe closest to the current position and eliminate the earliest keyframe. Therefore, BA will be locked in a time window. The output will be excluded. This method is called "sliding window".

Based on this idea, this paper designs a sliding window optimization method that is suitable for the optimization of the frame factor graph in this paper. The selection of keyframes is based on the simple and effective principle. If the position and attitude change of the lidar at a certain time exceeds the threshold defined in this paper, the point cloud frame F_{i+1} at that time is selected as the keyframe. In the factor graph, the recently saved keyframe F_{i+1} is connected to a new robot state node. Lidar frames between two keyframes are excluded. The keyframe method can not only balance the design density and memory consumption, but also reserve a relatively compact factor graph for nonlinear real-time optimization. The selection of lidar position and rotation threshold is 1m and 10°.

The generation process of sub-keyframes of voxel map is as follows. This paper first implements a sliding window method to create a point cloud map containing a fixed number of lidar scans. On this basis, the coordinate system conversion of point cloud data scanned by lidar between two consecutive frames is not directly optimized, but the first n keyframes of the point cloud of this frame are extracted for pose estimation. These keys are called sub-keys. Then the sub-keyframe set $\{F_{i-n}, \ldots, F_i\}$ is converted to the world coordinate system M_i using the transformation matrix $\{T_{i-n}, \ldots, T_i\}$ associated with it. The point cloud data under these sub keyframes are unified in a coordinate system to generate a voxel map M_i. In the previous feature extraction step, two types of features are extracted. So M_i is composed of two sub-voxel maps, namely, the edge feature voxel map M_e^i and the plane feature voxel map M_e^p. The relationship between the point cloud frame input by the lidar and the voxel map is shown in formula (17)–(19).

$$M_i = \{M_i^e,\ M_i^p\} \tag{17}$$

$$M_i^e = {}'F_i^e \cup {}'F_{i-1}^e \cup \ldots \cup {}'F_{i-n}^e \tag{18}$$

$$M_i^p = {}'F_i^p \cup {}'F_{i-1}^e \cup \ldots \cup {}'F_{i-n}^p \tag{19}$$

Sliding window algorithm process. The first step is to add new variables into the factor graph system for optimization. The second step is to remove the old variables if the number of variables in the sliding window reaches a certain dimension. That is, when a certain number of point cloud frames are input, the oldest frame will be removed. The third step is that the SLAM system continuously circulates the first step and the second step.

Using the keyframe as the reference variable in the factor graph will also increase the amount of calculation as the number of point cloud frames obtained increases. Therefore, when the number of variables in the sliding window reaches a certain number, the new variable will be obtained and the corresponding number of old variables will be removed. Here, the concept of the edge is applied to remove the old variables, and the information will be lost if the variables or the constraint relationship between the two frame variables are discarded directly. Therefore, the concept of the edge is used to transfer the information in the point cloud frame to be removed from the sliding window to the remaining point cloud frames in the sliding window using the method of marginal probability.

Edge in VINS is the oldest frame or the next new frame in the sliding window. Some old or unsatisfactory visual frames in the sliding window are eliminated. Therefore, the edge is also described as the process of decomposing the joint probability distribution into edge probability distribution and conditional probability distribution, that is, the process of using Schur method to supplement and reduce the optimization parameters.

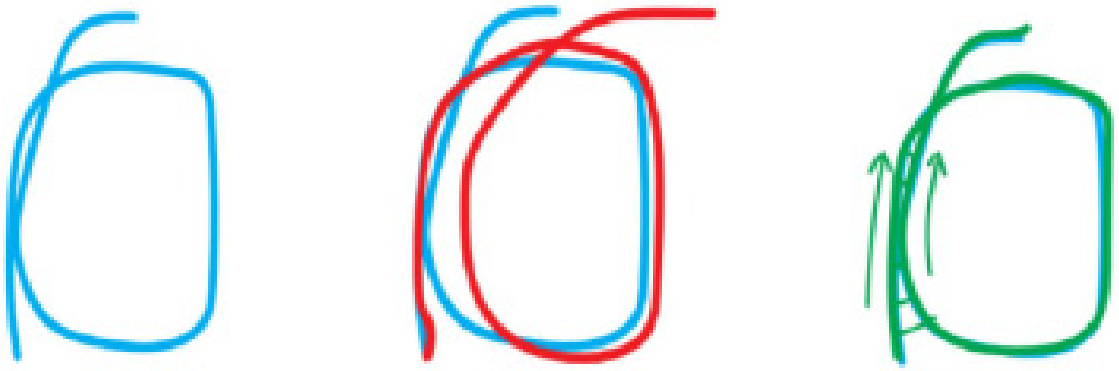
When a new frame is added, the old frame needs to be deleted. The goal of this paper is to use the edging is not to calculate the pose of this frame anymore, or even not to calculate some plane points and edge points related to this frame, but we hope to retain the constraint relationship of this frame to other frames in the sliding window, which will not reduce the constraint information, and is conducive to the optimization of state variables. When moving out the pose or feature, It is necessary to transform the associated constraint into a constraint item and focus on the optimization problem, which is what marginalization needs to do. Otherwise, when the old frame is moved out of the window, all the constraints associated with it will be lost, which will lead to a decline in the accuracy of the solution, and when the robot is in degenerate motion (such as uniform motion), it cannot be solved without historical information as constraints.

3.2. Research on the loopback detection algorithm

Based on the improved mapping method optimized by factor graph, the Scan Context loopback detection method is integrated. For maps with loopback, the fusion method can better improve the mapping accuracy. When loopback is detected, the position and attitude of the laser radar will be optimized. The position and pose are introduced as loopback detection factors into the factor diagram designed in this paper, which combines IMU and lidar factors. As the third factor, the optimization effect of the factor diagram is improved. The experiment shows that under the map with loopback, the mapping accuracy is improved and the matching speed between two frames is reduced, but the real-time positioning and mapping are not affected, which can meet the requirements of real-time positioning and mapping in practical applications.

3.2.1. Loop back detection Scan Context

The estimation of the front-end pose has accumulated errors, which can be partially reduced by optimizing the back end method. However, in practice, due to the practicability of the system, not all data can be optimized during final optimization, so it is impossible to completely eliminate accumulated errors, which accumulate continuously during system operation. The loopback detection aims to correct the global position and posture through the visual constraints between the current frame and the historical frame, reduce the accumulated errors, and provide more effective data for the final optimization when the robot determines the previously checked area. Figure 2 shows the study of loopback detection to eliminate errors.



a) Truth track b) No loopback track c) Loop track

Figure 2. Error elimination schematic diagram of loop detection.

Loopback test is a relatively independent classic SLAM framework, mainly when robots are readjusted in some scenes, they may generate additional constraints on the existing historical track through optimization, and correlate with the current position to correct the relevant historical track changes to reduce cumulative errors. Figure 3 shows the schematic diagram of loopback detection, and the small bar in the diagram represents the track point. The red bar indicates the existence of the loopback. Use this constraint relationship to correct the trajectory to obtain the results shown in the right Figure 3.

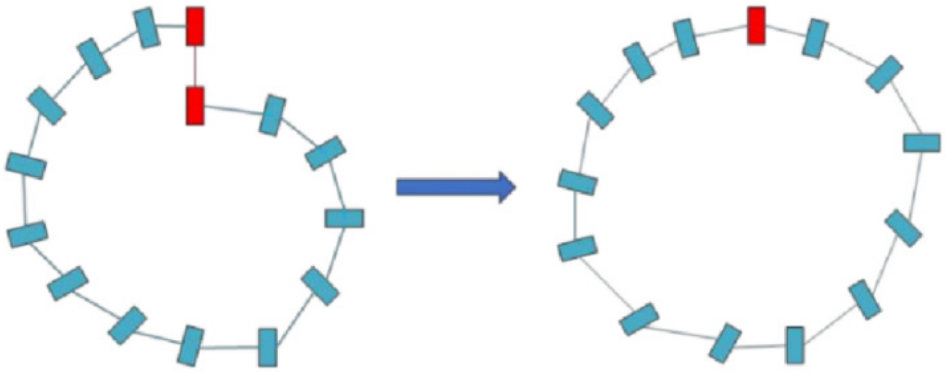


Figure 3. Loopback detection.

The SLAM framework in this paper integrates the Scan Context loopback detection algorithm proposed by Giseop Kim and Young Kim as the loopback detection algorithm in this paper to reduce the cumulative error, and integrates the parameters provided by Scan Context into the factor graph as closed-loop factors, providing more effective data for back-end optimization. The algorithm is shown in Table 1.

Table 1. Fusion Scan Context pseudocode.

Fusion Scan Context pseudocode
Input：
raw data：*laserCloud;*
Output：
Match data：*laserMatchCloud;*
1 Generating function Scan context;
2 *makeScancontext（laserCloud）;*
3 Projected into the radar coordinate system;
4 if *havelaserCloud* then
5 $azimrange = \sqrt{laserCloud.x^2, laserCloud.y^2, laserCloud.xz^2}$;
6 $azimrangle = \theta(laserCloud.x, laserCloud.y)$;
7 end
8 invariantKey = makeRingkeyFromScancontext ();
9 variantKey = makeSectorkeyFromScancontext ();
10 Kdtree→index→find (result);
11 distanceBtnScanContext (laserCloud,curlaserCloud);
12 Output:laserMatchCloud;

3.2.2. Algorithm fusion principle

First, the point cloud in the 3D scan is encoded into the matrix. Then, the Nr (ring number) dimension vector is encoded from the matrix and used to retrieve the candidate object matrix. At the same time, the current matrix is constructed by the KD tree. Finally, compare the similarity between the retrieved candidate object and the current matrix, and the candidate that meets the threshold and is closest to the current matrix is considered a closed loop.

4. Result analysis

4.1. Factor graph optimization experiment and analysis

On the experimental equipment, the NVIDIA GeForce GTX2060 graphics card with 16GB memory is used uniformly. And the LINUX16.04 operating system on the Intel Core i7-9750H CPU. In terms of the experimental environment, the robot operating system ROS is used uniformly to run simultaneous localization and mapping algorithms.

4.1.1. Experimental parameters

In terms of experimental parameters, some experimental parameters set in the experiment are

selected. Table 2. shows the gravity influence parameters. When IMU calculates the speed of an object, it will add the gravitational acceleration of the object by default. The parameters in this table will be used to remove the influence of gravity on velocity. Table 3. is the table of edge parameters. When using the edge method to remove the old frames in the sliding window, the Shure compensation method can achieve the best effect by adjusting the parameters in the table. Table 4. shows the IMU noise parameters. The noise parameters of IMU need to be measured before using IMU for experiments. The error of IMU measurement data is eliminated by establishing IMU noise model and using IMU noise parameters.

Table 2. Gravity effect.

Parameter name	Parameter value
update_laser_imu	1
gravity_fix	1
plane_projection_factor	0
imu_factor	1
point_distance_factor	1
prior_factor	0

Table 3. Marginalized.

Parameter name	Parameter value
marginalization_factor	1
odom_io	0
pcl_viewer	0

Table 4. IMU noises.

Parameter name	Parameter value
acc_n	0.2
gyr_n	0.02
acc_w	0.0002
gyr_w	2.0e-5
g_norm	9.805

4.1.2. Result analysis

1) Comparison of drawing effects

On the outdoor data set, in order to verify the changes in the sliding window and edge probability used in this paper. The mapping methods of using the sliding window and factor graph of edge probability will be compared experimentally. As shown in Figure 4 below, the left side shows the drawing effect without sliding window and edge probability method. On the right is the mapping effect using sliding window and edge probability method. The red arrow in the figure indicates the change of the position and attitude of the lidar in each frame. By comparing the transformation of radar pose in the two figures. It can be seen that the use of sliding windows and edge probability does not reduce the estimation effect of radar pose, but improves the mapping effect. There is less noise in the right image than in the left image. This is due to the use of sliding windows and edge methods. This increases

the weight ratio of the influence of the front and rear frames near the input frame on the frame.

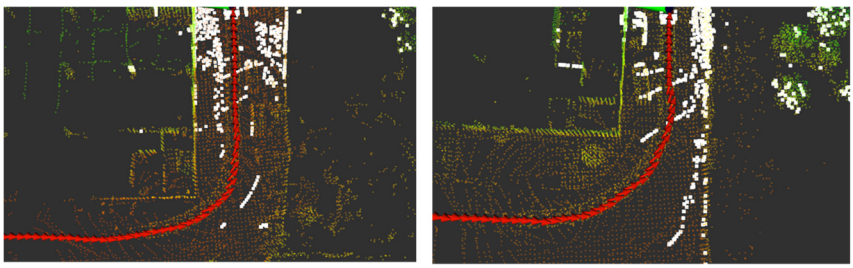
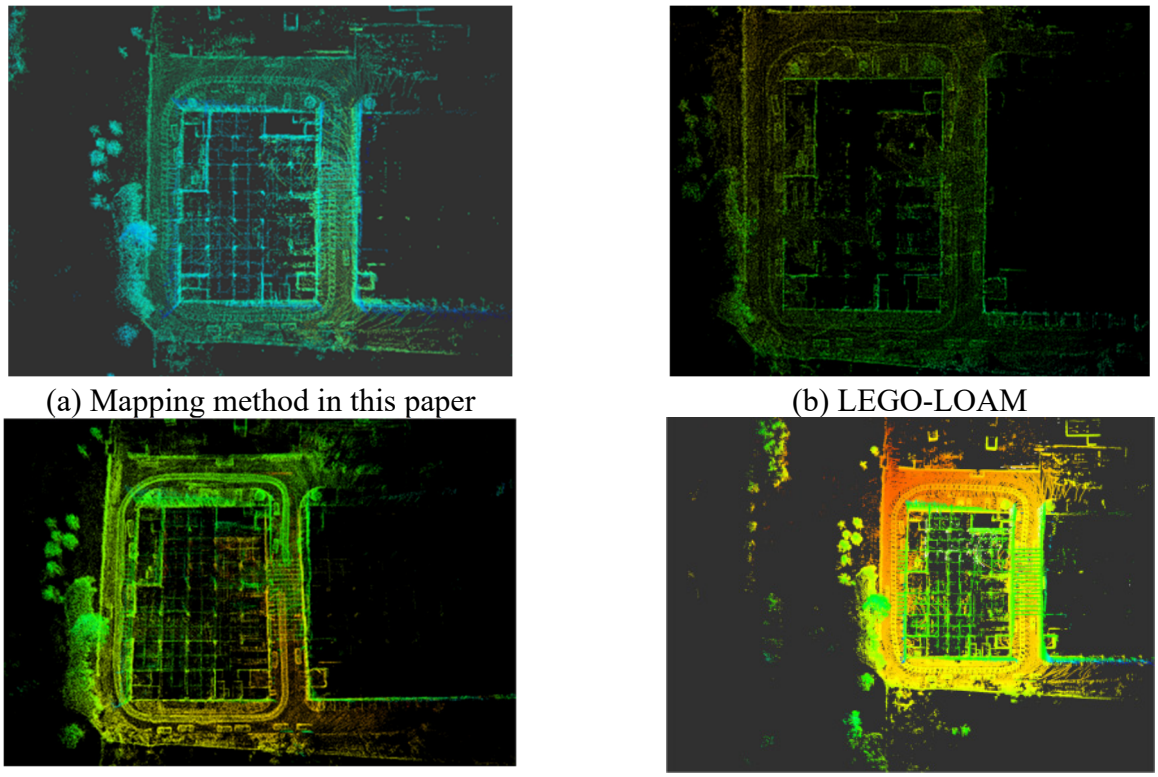


Figure 4. Effect comparison of drawing construction.

LOAM is a point cloud matching algorithm used by the front end to locate and map at the same time with a good outdoor effect. LEGO-LOAM is an algorithm with excellent outdoor mapping effects when using lidar and IMU sensors. Therefore, this paper compares these two algorithms. Figure 5 compares the mapping results of the algorithm in this paper with the open-source algorithm LOAM and LEGO-LOAM in the outdoor environment. Figure 5(a) What is shown is the mapping effect of this paper. The mapping result of LEGO-LOAM algorithm is shown in Figure 5(b). It can be seen that the mapping result of LEGO-LOAM in the loose coupling mode of LIDAR and IMU is obviously inferior to that proposed in this paper. The mapping result of the LOAM algorithm is shown in Figure 5(c). It can be seen from the mapping results that there are too many noise points on the algorithm LOAM map. The effect of LIO-SAM algorithm mapping is shown in Figure 5(d), which is similar to that in this paper.



(a) Mapping method in this paper (b) LEGO-LOAM

(c) LOAM (d) LIO-SAM

Figure 5. Comparison of the results of four algorithms.

2) Comparison of drawing effect error

Figure 6 shows the error of the algorithm proposed in this paper, the LOAM algorithm and the LEGO-LOAM algorithm in X, Y and Z directions estimated using the evo evaluation tool. In this paper, the GPS data is used as the true value for error comparison. The green line represents the algorithm error proposed in this paper. The purple line represents the algorithm error of LEGO-LOAM. The red line represents the LOAM algorithm error. The blue line represents the algorithm error of LIO-SAM. It can be seen from Figure 6 that the error of the algorithm proposed in this paper is smaller than that of LEGO-LOAM and LOAM on the XYZ axis. Table 5. shows the comparison between the improved factor graph algorithm using sliding windows and the average error of other algorithms. It can be seen that the mapping error on the XYZ axis is greatly reduced compared with the point line registration method without the back-end before improvement. Compared with LEGO-LOAM, the error is significantly improved. There is little difference with LIO-SAM error.

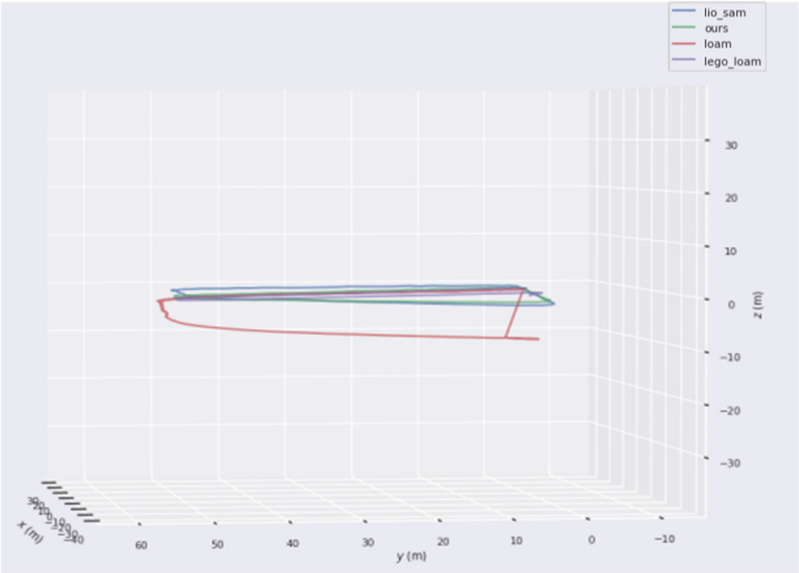


Figure 6. Error comparison.

Table 5. Comparison of trajectory mean error.

algorithm	X axis(m)	Y axis(m)	Z axis(m)
LOAM	3.87	2.98	3.4
LEGO-LOAM	1.85	2.21	1.98
LIO-SAM	0.79	0.80	0.72
Registration method without back-end point line	3.54	3.10	2.99
Methods in this paper	0.81	0.78	0.74

Table 6 shows the comparison of the average angle error between the algorithm using the improved factor graph of the sliding window and other algorithms. The pitch is the rotation around the X-axis, which is called pitch angle. Yaw is the rotation around the Y-axis, called the yaw angle. The roll is rotated around the Z-axis, called the roll angle. It can be seen from Table 6 that the algorithm in this paper is better than other algorithms, and has little difference from LIO-SAM.

Table 6. Comparison of angle mean error.

algorithm	pitch(deg)	roll(deg)	yaw(deg)
LOAM	16.2	15.3	11.4
LEGO-LOAM	13.4	13.8	13.4
LIO-SAM	2.5	2.9	8.2
Registration method without back-end point line	15.8	14.2	13.9
Methods in this paper	2.6	2.8	8.5

3) Running speed, point cloud quantity comparison

The average registration time between every two frames obtained by using the sliding window and edge probability optimization factor graph is shown in Table 7 below. The average registration time of the factor graph algorithm optimized by sliding window and edge probability is 12% higher than that of the algorithm without sliding window. Reduce the number of point cloud frames in the incoming factor graph by using keyframes. You can reduce the number of point clouds on the map without affecting map creation. As shown in Table 8, the number of point clouds decreases by about 70% and the storage space decreases by 69% after using keyframes.

Table 7. Comparison of algorithm running speed.

algorithm	Average registration time(ms)
Use window sliding and edge probability	47.2
Do not use window sliding and edge probability	54.6

Table 8. Point cloud number and storage space comparison.

algorithm	Number of point clouds	Storage space (MB)
Use Keys	121582	98.4
Don’t use keyframes	173688	142

4.2. Loop detection experiment and analysis

4.2.1. Experimental parameters

This environment uses the Linux+ROS platform. Some experimental parameters of this experiment are as follows. Table 9 shows the parameter values under the point cloud data processing method collected by the laser radar. The configuration is divided into two parts. The first part is the radar configuration, including the minimum vertical resolution and the minimum matching distance. The other part is the parameter setting of filtering, which is the parameter for processing the number of point clouds retained per unit area. Table 10 is the parameters of the sliding window size. Set the sliding window size by setting the parameters in the table, and use the frame of the sliding window to compare with the new point cloud frame. Table 11 shows the parameters of the external parameter rotation matrix. Setting the external parameter rotation matrix can make the IMU and the laser radar consistent in the angle deflection, which is convenient for the creation of the map in the experiment.

Table 9. Lidar features.

Parameter name	Parameter value
min_plane_dis	0.2
min_match_sq_dis	1.0
corner_filter_size	0.2
surf_filter_size	0.4
map_filter_size	0.6

Table 10. Sliding window size.

Parameter name	Parameter value
window_size	12
opt_window_size	7
init_window_factor	2

Table 11. External parameter rotation matrix.

Parameter name	Parameter value
extrinsic_rotation	!!opencv-matrix1
rows	3
cols	3
dt	d
data	[1, 0, 0, 0, 1, 0, 0, 0, 1]

4.2.2. Result analysis

The experimental environment is the same as that of the previous experiment. First, the point cloud map is built using the mapping method that incorporates the loopback detection method, and then the map is built using the same data set using the mapping method that does not add the loopback detection method.

1) Comparison of drawing effects

Figure 7 shows the point cloud map of two angles without the loopback detection method. Comparing the point cloud map of two angles with the loopback detection method in Figure 8, it can be found that in some details, the point cloud map with the loopback detection method is better than the point cloud map without the loopback detection method.

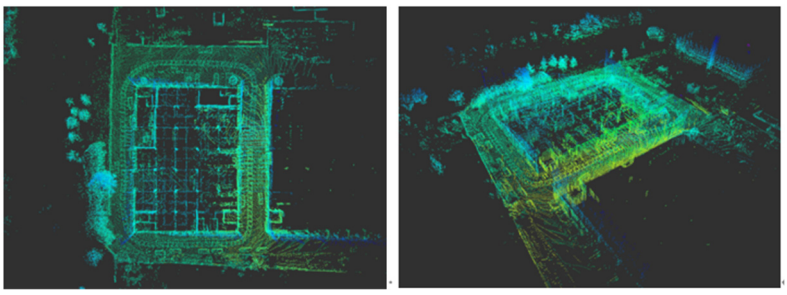


Figure 7. Point cloud map without loopback detection.

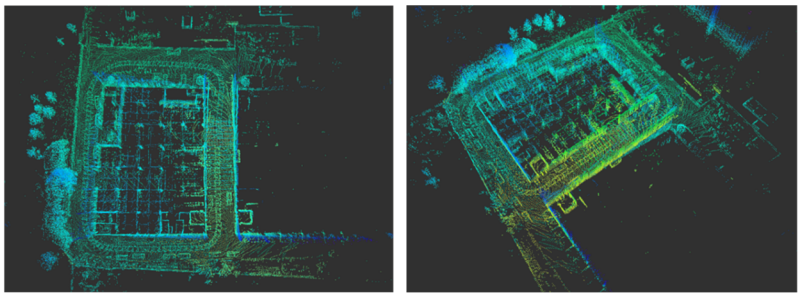


Figure 8. Point cloud map with loopback detection.

2) Comparison of drawing effect error

Table 12 compares the average error of the maps built before and after the fusion loop detection algorithm in the three directions of the XYZ axis. It can be clearly seen that the error is reduced after the loopback detection algorithm is fused.

Table 12. Comparison of trajectory mean error.

algorithm	X-axis(m)	Y-axis(m)	Z-axis(m)
Before fusion loopback detection algorithm	0.81	0.78	0.74
After fusion of loopback detection algorithm	0.68	0.71	0.67

Then, the evo evaluation tool is used to compare the error of the mapping method using loopback detection and the mapping method without loopback detection in the X, Y and Z axes. Figure 9 shows the changes of three of the two algorithms in the process of laser radar mapping, and compares the changes of angle roll on the X-axis rotation angle pitch, Y-axis rotation angle yaw, and Z-axis rotation angle with or without loopback detection in the process of mapping. Table 13 shows that the average error of pitch, yaw and roll is better than the algorithm without fusion loopback detection.

Figure 9. Euler Angle change contrast.

Table 13. Angle mean error comparison.

algorithm	pitch(deg)	roll(deg)	yaw(deg)
Before fusion loopback detection algorithm	2.6	2.8	8.5
After fusion of loopback detection algorithm	2.1	2.2	5.5

3) Operating speed comparison

Table 14 compares the running speed before and after the fusion loopback detection algorithm. The fusion loopback detection algorithm will slightly reduce the matching speed between the two frame point clouds, but the reduced speed will not affect the simultaneous positioning and mapping.

Table 14. Running speed comparison.

algorithm	Average registration time(ms)
Before fusion loopback detection algorithm	47.2
After fusion of loopback detection algorithm	52.8

5. Conclusions

This paper mainly studies the algorithm of simultaneous positioning and mapping of robots based on lidar in an outdoor environment, focusing on optimizing the global position and pose through factor graph, and using a window sliding method to improve the optimization speed of the factor graph, and adding loopback detection module to deal with the problem of large-scale mapping. The experiment shows that the back-end optimization method using the window sliding method and edge probability can significantly improve the optimization speed and basically do not lose the optimization accuracy. The speed is increased by 12%, and the effect and error of image construction are significantly better than the two algorithms, LOAM and LEGO-LOAM, and the effect of LIO-SAM is not much different. In addition, a fusion method is proposed, using Scan Context as the loopback detection method in this paper, and experimental comparison is carried out. The results show that the loopback detection method has a better effect and smaller error.

Acknowledgments

This work was supported by Joint Fund Project of the National Natural Science Foundation of China (U1908218), the Natural Science Foundation project of Liaoning Province (2021-KF-12-06), and the Department of Education of Liaoning Province (LJKFZ20220197).

References

1. L. C. van der Gaag, J. Ch. Meyer, Informational independence: Models and normal forms, *Int. J. Intell. Syst.*, **13** (1998), 83–109. https://doi.org/10.1002/(SICI)1098-111X(199801)13:1<83::AID-INT7>3.3.CO;2-F
2. R. Yagfarov, M. Ivanou, I. Afanasyev, Map comparison of lidar-based 2D slam algorithms using precise ground truth, in *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, (2018), 1979–1983. https://doi.org/10.1109/ICARCV.2018.8581131

3. R. Smith, M. Self, P. Cheeseman, Estimating uncertain spatial relationships in robotics, in *1987 IEEE International Conference on Robotics and Automation*, (1987), 850. https://doi.org/10.1109/ROBOT.1987.1087846
4. A. Doucet, N. D. Freitas, K. Murphy, S. Russell, Rao-Blackwellised particle filtering for dynamic Bayesian networks, preprint, arXiv:1301.3853.
5. M. Montemarlo, S. Thrun, D. Koller, B. Wegbreit, FastSLAM: A factored solution to the simultaneous localization and mapping problem, in *Proceedings of the AAAI National Conference on Artificial Intelligence*, (2002).
6. M. Montemerlo, S. Thrun, Simultaneous localization and mapping with unknown data association using FastSLAM, in *2003 IEEE International Conference on Robotics and Automation*, **2** (2003), 1985–1991. https://doi.org/10.1109/ROBOT.2003.1241885
7. G. Grisetti, C. Stachniss, W. Burgard, Improved techniques for grid mapping with Rao-Blackwellized particle filters, *IEEE Trans. Rob.,* **23** (2007), 34–46. https://doi.org/10.1109/TRO.2006.889486
8. J. Zhang, S. Singh, Low-drift and real-time lidar odometry and mapping, *Auton. Robot.*, **41** (2017), 401–416. https://doi.org/10.1007/s10514-016-9548-2
9. W. Hess, D. Kohler, H. Rapp, D. Andor, Real-Time Loop Closure in 2D LIDAR SLAM, in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, **9** (2016), 1271–1278. https://doi.org/10.1109/ICRA.2016.7487258
10. T. Shan, B. Englot, 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)*, (2018), 4758–4765. https://doi.org/10.1109/IROS.2018.8594299
11. D. Droeschel, S. Behnke, Efficient continuous-time SLAM for 3D lidar-based online mapping, in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, (2018), 16710–16728. https://doi.org/10.1109/ICRA.2018.8461000
12. J. Behley, C. Stachniss, Efficient surfel-based SLAM using 3D laser range data in urban environments, *Rob.: Sci. Syst.*, (2018). https://doi.org/10.15607/RSS.2018.XIV.016
13. M. Li, H. Zhu, S. You, C. Tang, Y. Li, Efficient laser-based 3D SLAM for coal mine rescue robots, in *2018 IEEE 8th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*, (2018), 971–976. https://doi.org/10.1109/CYBER.2018.8688349
14. J. E. Deschaud, IMLS-SLAM: Scan-to-model matching based on 3D data, in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, (2018), 2480–2485. https://doi.org/10.1109/ICRA.2018.8460653
15. W. Shao, S. Vijayarangan, C. Li, G. Kantor, Stereo visual inertial LiDAR simultaneous localization and mapping, in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (2019), 370–377. https://doi.org/10.1109/IROS40897.2019.8968012
16. F. Pang, S. Wei, X. Shi, K. Chen, Milometer and mapping method for inertial navigation coupling of lidar, *Appl. Res. Comput.*, **38** (2021), 1–7. https://doi.org/10.19734/j.issn.1001-3695.2020.06.0259
17. G. Niu, Y. Wang, Unmanned vehicle positioning and mapping method based on multi- constraint factor graph optimization, *J. Beijing Univ. Aeronaut. Astronaut.*, **47** (2021), 306–314. https://doi.org/10.13700/j.bh.1001-5965.2020.0212

18. T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, D. Rus, Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping, in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (2020), 5135–5142. https://doi.org/10.1109/IROS45743.2020.9341176
19. X. Ma, X. Yao, L. Ding, T. Zhu, Variable motion model for lidar motion distortion correction, in *AOPC 2021: Optical Sensing and Imaging Technology*, **1206524** (2021), 489–495. https://doi.org/10.1117/12.2606143
20. J. Wang, H. Zhu, H. Liu, Z. Ma, Lossy point cloud geometry compression via end-to-end learning, *IEEE Trans. Circuits Syst. Video Technol.*, **31** (2021), 4909–4923. https://doi.org/10.1109/TCSVT.2021.3051377
21. W. Sun, J. Wang, F. Jin, Y. Yang, A quality improvement method for 3D laser slam point clouds based on geometric primitives of the scan scene, *Int. J. Remote Sens.*, **42** (2021), 378–388. https://doi.org/10.1080/2150704X.2020.1811911
22. M. Li, H. Zhu, S. You, C. Tang; Y. Li, Efficient laser-based 3D SLAM for coal mine rescue robots, in *2018 IEEE 8th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*, (2018), 971–976. https://doi.org/10.1109/CYBER.2018.8688349
23. X. Zhang, G. Lu, G. Fu, D. Xu, S. Liang, SLAM algorithm analysis of mobile robot based on lidar, in *2019 Chinese Control Conference (CCC)*, (2019), 4739–4745. https://doi.org/10.23919/ChiCC.2019.8866200
24. D. V. Nam, K. Gon-Woo, Solid-state LiDAR based-SLAM: A concise review and application, in *2021 IEEE International Conference on Big Data and Smart Computing (BigComp)*, (2021), 302–305. https://doi.org/10.1109/BigComp51126.2021.00064
25. N. Akiyama, S. Nishiyama, K. Sakita, J. Song, F. Yamazaki, River levees monitoring using three dimensional laser point clouds with SLAM technology, in *Civil Infrastructures Confronting Severe Weathers and Climate Changes Conference*, (2021), 14–22. https://doi.org/10.1007/978-3-030-79798-0_2
26. J. Liu, Q. Sun, Z. Fan, Y. Jia, TOF lidar development in autonomous vehicle, in *2018 IEEE 3rd Optoelectronics Global Conference (OGC)*, (2018), 185–190. https://doi.org/10.1109/OGC.2018.8529992
27. J. Lv, J. Xu, K. Hu, Y. Liu; X. Zuo, Targetless calibration of lidar-imu system based on continuous-time batch estimation, in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (2020), 9968–9975. https://doi.org/10.1109/IROS45743.2020.9341405

AIMS Press

©2023 the Author(s), licensee AIMS Press. This is an open access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0)