Initial Position and Orientation Estimation of Scan Matching Based on the Height Feature Map

Yu-Xiang Long
School of Mechanical and Electrical Engineering
Guangzhou University
Guangzhou, 51006, China
1226404425@qq.com

Zhi-Fu Li*
School of Mechanical and Electrical Engineering
Guangzhou University
Guangzhou, 51006, China
lizhifu8@163.com

2024 9th International Conference on Automation, Control and Robotics Engineering (CACRE) | 979-8-3503-5030-2/24/$31.00 ©2024 IEEE | DOI: 10.1109/CACRE62362.2024.10635017

***Abstract*—In order to solve the problem that the uniform speed model of lidar odometer can't accurately predict the location at variable speed, the method based on height feature map is proposed in this paper, which extracts co-visible regions between adjacent frames and matches them in feature grid maps to obtain initial position and orientation. By testing it on the kitti data and compared it with F-LOAM, this method shows better accuracy in trajectory and improves the stability of the algorithm at low iteration.**

Keywords—lidar odometer, 3D SLAM, initial location

I. Introduction

With the development of autonomous driving and the increasing demand, SLAM(Simultaneous Localization And Mapping) shows broader prospect and valuable research significance. Nowadays, SLAM primarily revolves around two core sensors: lidar and camera, which leading to the development of single-sensor SLAM or multi-sensor fusion SLAM.

Lidar SLAM is divided into 2D SLAM and 3D SLAM. 2D lidar SLAM has developed from the initial filtering scheme to the current optimized scheme, and the positioning accuracy and mapping effect have been greatly improved. A milestone in this field is the Cartographer [1] algorithm proposed by Google, which presents a complete Lidar SLAM solution based on graph optimization. Its correlation scan matching method offers improved point cloud registration in dynamic environments.

Among 3D Lidar SLAM methods, pure Lidar SLAM is represented by the LOAM [2](Lidar Odometry and Mapping) series, which proposes a point cloud registration scheme based on point cloud features. This approach significantly improves registration speed and enables real-time processing of 3D point clouds. Point cloud feature extraction not only streamlines massive point cloud data but also enhances the expressiveness of the processed data regarding the environment, leading to better performance in non-degraded scenarios.

A-LOAM extracts line features, surface features, corner points, and surface points from adjacent point clouds. It then constructs residuals based on the distances from corner points to line features and the distances between surface points and surface features. The optimization variables involve displacement vectors and XYZ axis rotation angles and iterative calculations yield optimized and accurate poses.

Based on A-LOAM, F-LOAM [3] proposes a "two-step method" to de-distort the point cloud, improving the time consuming iterative de-distortion process. Furthermore, Lie group Lie algebra is used to construct optimization variables, which optimizes the solution space and significantly improves location accuracy and speed.

LeGO-LOAM [4] is a SLAM algorithm designed for uneven ground and incorporates loop closure detection to optimize uneven ground scenarios while ensuring lightweight processing. The development of loop closure detection makes the Lidar SLAM more effective [5].

In summary, the above methods mainly optimize the feature extraction, the process of nonlinear optimization and the construction ground constraints, but are insufficient for initial position and orientation of nonlinear optimization. Therefore, feature matching based on the height feature map [6] is used to estimate the initial position and orientation. Firstly, the height of the lidar point cloud is used as the feature map. Secondly, the common view between two feature map is obtained. Thirdly, By comparing the difference between common views, the initial position and orientation are obtained. Finally these initial values are used in ICP(iterative nearest point) [7] instead of the uniform motion model.

II. Methodology

A. Pose Calculation Based on 3D Lidar

Lidar SLAM is usually positioned by point cloud registration. One of the most commonly used solutions is ICP, which recursively obtains the robot's pose in the world coordinate system. LOAM proposes point cloud matching based on point cloud features on point-to-point ICP, which not only improves the operation efficiency but also improves the solution accuracy.

Scan-to-scan only uses the point clouds of two adjacent scans for matching, but this method uses too little point cloud information, resulting in low pose accuracy. Therefore, a map composed of point clouds within a certain range is used as the matching object, and the current scan is used for matching with it to get the pose by scan-to-map.

The pose transformation between two consecutive scans of the 3D point cloud can be represented by a homogeneous transformation matrix $\boldsymbol{T}$. Assuming that the 3D lidar displacement and rotates at a constant speed, then the initial value of the current lidar pose can be deduced by linear interpolation based on the previously obtained pose change. Define $\boldsymbol{T}_K^W$ as the pose of the k^{th} scan in the world coordinate system, then the initial pose estimation of the k^{th} scan can be expressed as

$$\boldsymbol{T}_{\boldsymbol{k0}}^{\boldsymbol{W}} = \boldsymbol{T}_{\boldsymbol{k-1}}^{\boldsymbol{W}}\left(\boldsymbol{T}_{\boldsymbol{k-2}}^{\boldsymbol{W-}} \cdot \boldsymbol{T}_{\boldsymbol{k-1}}^{\boldsymbol{W}}\right) \tag{1}$$

$\boldsymbol{P}_{(\boldsymbol{k},\boldsymbol{i})}^{\boldsymbol{B}}$ is the i^{th} point of the k^{th} scan in the lidar coordinate system, then the coordinates of this point in the world coordinate system can be expressed as (2).

$$\boldsymbol{P}_{(\boldsymbol{k},\boldsymbol{i})}^{\boldsymbol{W}} = \boldsymbol{T}_{\boldsymbol{k}}^{\boldsymbol{W}} \cdot \boldsymbol{P}_{(\boldsymbol{k},\boldsymbol{i})}^{\boldsymbol{B}} \tag{2}$$

When multiplied with the homogeneous transform matrix, $\boldsymbol{P}_{(\boldsymbol{k},\boldsymbol{i})}^{\boldsymbol{B}}$ is augmented into a vector with four rows and one column, where the last element is 1.

Define $\boldsymbol{M}_{\mathbf{0:k}}^{\boldsymbol{W}}$ as the point cloud geometry from the initial moment to the current moment in the world coordinate system, d_E and d_H are respectively the distance from the corner point to the corresponding edge line and the distance from the plane feature point to the corresponding plane distance:

$$f_E\left(\boldsymbol{P}_{(\boldsymbol{k},\boldsymbol{i})}^{\boldsymbol{W}}, \boldsymbol{M}_{\mathbf{0}:\boldsymbol{k}}^{\boldsymbol{W}}\right) = d_E \tag{3}$$
$$f_E\left(\boldsymbol{P}_{(\boldsymbol{k},\boldsymbol{i})}^{\boldsymbol{W}}, \boldsymbol{M}_{\mathbf{0}:\boldsymbol{k}}^{\boldsymbol{W}}\right) = d_H \tag{4}$$

The overall cost function can be unified as (5).

$$f_E\left(\boldsymbol{P}_{(\boldsymbol{k},\boldsymbol{i})}^{\boldsymbol{W}}, \boldsymbol{M}_{\mathbf{0}:\boldsymbol{k}}^{\boldsymbol{W}}\right) = d_H + d_E = \boldsymbol{d} \tag{5}$$

The objective function of the ICP algorithm for point-line-surface features can be expressed as (6).

$$\min_{\boldsymbol{T}_{\boldsymbol{k}}^{\boldsymbol{W}}} \frac{1}{2} d^2 \tag{6}$$

Since the Lie group $\boldsymbol{T}$ does not satisfy the addition operation in operation, the iterative calculation of the pose increment cannot be performed. Instead, the Lie algebra $\boldsymbol{\varepsilon}$ is used for the incremental solution at the optimization calculation. The mapping relationship between $\boldsymbol{T}$ and $\boldsymbol{\varepsilon}$ can be simply expressed as a logarithmic relationship:

$$ln\boldsymbol{T} = \boldsymbol{\varepsilon} \tag{7}$$

The original target function is equivalent to:

$$\min_{\varepsilon_{\boldsymbol{k}}^{W}} \frac{1}{2} \boldsymbol{d}^2 \tag{8}$$

The Jacobian matrix **J** is:

$$\mathbf{J} = \frac{\partial \boldsymbol{d}}{\partial \varepsilon_{\boldsymbol{k}}^{\boldsymbol{W}}} \tag{9}$$

Solve the incremental equation using the Gauss-Newton:

$$\mathrm{H}\left(\boldsymbol{\varepsilon}_{k}^{W}\right)\Delta\boldsymbol{\varepsilon}_{k}^{W}=\mathrm{g}\left(\boldsymbol{\varepsilon}_{k}^{W}\right) \tag{10}$$

where $\mathbf{H}\left(\boldsymbol{\varepsilon}_{\mathbf{k}}^{\mathbf{W}}\right)$ and $\mathbf{g}\left(\boldsymbol{\varepsilon}_{\mathbf{k}}^{\mathbf{W}}\right)$ are (11)、(12).

$$\mathrm{H}\left(\boldsymbol{\varepsilon}_{k}^{W}\right)=\mathrm{J}\left(\boldsymbol{\varepsilon}_{k}^{W}\right)\mathrm{J}\left(\boldsymbol{\varepsilon}_{k}^{W}\right)^{\mathrm{T}} \tag{11}$$
$$\mathrm{g}\left(\boldsymbol{\varepsilon}_{k}^{W}\right)=-\mathrm{J}\left(\boldsymbol{\varepsilon}_{k}^{W}\right)\cdot\boldsymbol{d} \tag{12}$$

Solve the pose of the current lidar scan represented by Lie algebra in the world coordinate system:

$$\boldsymbol{\varepsilon}_{\boldsymbol{k}}^{\boldsymbol{W}} = \boldsymbol{\varepsilon}_{\boldsymbol{k}}^{\boldsymbol{W}} + \boldsymbol{\Delta\varepsilon}_{\boldsymbol{k}}^{\boldsymbol{W}} \tag{13}$$

Then calculate the pose $\boldsymbol{T}$ of the current scan through the exponential mapping relationship (14).

$$\boldsymbol{e}^{\boldsymbol{\varepsilon}} = \boldsymbol{T} \tag{14}$$

B. Extract The Feature of Similar Point Cloud

The original point cloud data contains a large number of points, and the feature extraction can reduce the amount of data while enhancing the features of the point cloud, without losing accuracy.

The extraction of feature points from the original point cloud data is mainly to obtain the corner points and surface points. A corner point is a point where the curvature changes greatly, which can be defined by equation (15).

$$c_{(k,i)} = \frac{1}{2n}\sum_{j=-n,j\neq 0}^{n}\left\|\left(\boldsymbol{P}_{(\boldsymbol{k},\boldsymbol{i})}^{\boldsymbol{B}} - \boldsymbol{P}_{(\boldsymbol{k},\boldsymbol{j})}^{\boldsymbol{B}}\right)\right\| \tag{15}$$

Use the above point cloud features to form the point-line residual function and the point-line residual function respectively:

$$d_E = \frac{\left|\left(\boldsymbol{E}_{(\boldsymbol{k},\boldsymbol{i})}^{\boldsymbol{B}} - \boldsymbol{E}_{(\boldsymbol{k-1},\boldsymbol{j})}^{\boldsymbol{B}}\right) \times \left(\boldsymbol{E}_{(\boldsymbol{k},\boldsymbol{i})}^{\boldsymbol{B}} - \boldsymbol{E}_{(\boldsymbol{k-1},\boldsymbol{l})}^{\boldsymbol{B}}\right)\right|}{\left|\boldsymbol{E}_{(\boldsymbol{k-1},\boldsymbol{j})}^{\boldsymbol{B}} - \boldsymbol{E}_{(\boldsymbol{k-1},\boldsymbol{j})}^{\boldsymbol{B}}\right|} \tag{16}$$

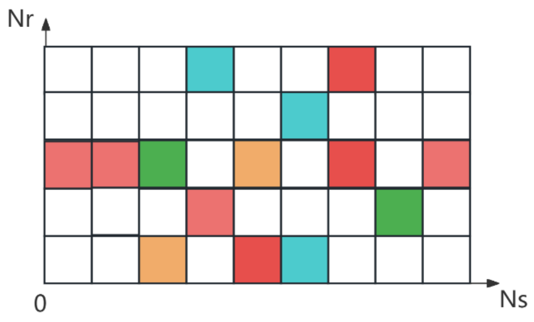
$\boldsymbol{E}$ is a point set containing only edge points. Its subscript k denotes the current scan, $k-1$ represents the previous scan. i is the edge point of the current scan, and j and l are the points on the two scanning lines closest to i in the previous scan.

The point-plane residual function is (17).

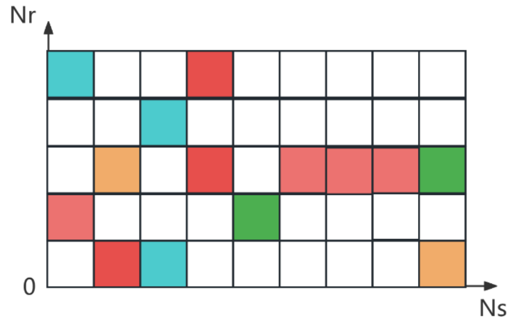
$$d_H = \frac{\left(\boldsymbol{H}_{(\boldsymbol{k},\boldsymbol{i})}^{\boldsymbol{B}} - \boldsymbol{H}_{(\boldsymbol{k-1},\boldsymbol{j})}^{\boldsymbol{B}}\right) \cdot \left(\begin{array}{c}\left(\boldsymbol{H}_{(\boldsymbol{k-1},\boldsymbol{j})}^{\boldsymbol{B}} - \boldsymbol{H}_{(\boldsymbol{k-1},\boldsymbol{l})}^{\boldsymbol{B}}\right) \times \\ \left(\boldsymbol{H}_{(\boldsymbol{k-1},\boldsymbol{j})}^{\boldsymbol{B}} - \boldsymbol{H}_{(\boldsymbol{k-1},\boldsymbol{m})}^{\boldsymbol{B}}\right)\end{array}\right)}{\left|\left(\boldsymbol{H}_{(\boldsymbol{k-1},\boldsymbol{j})}^{\boldsymbol{B}} - \boldsymbol{H}_{(\boldsymbol{k-1},\boldsymbol{l})}^{\boldsymbol{B}}\right) \times \left(\boldsymbol{H}_{(\boldsymbol{k-1},\boldsymbol{j})}^{\boldsymbol{B}} - \boldsymbol{H}_{(\boldsymbol{k-1},\boldsymbol{m})}^{\boldsymbol{B}}\right)\right|} \tag{17}$$

$\boldsymbol{H}$ is a point set containing only surface points. Its subscript k denotes the current scan, $k-1$ represents the previous scan. i is the surface point of the current scan, and j and l are the points on the two scanning lines closest to i in the previous scan. m is the surface point closest to i, belonging to the two scanning lines mentioned above.

Using the 3D point sets $\boldsymbol{H}$ and $\boldsymbol{E}$, a 2D point cloud orientation grid map is constructed, which contains specific 3D information. The construction involves creating a circular fan graph in the robot coordinate system which is expanded like Fig. 1. The number of rings, denoted as N_r, is determined by the displacement resolution and the size of the circular fan graph. Similarly, the number of fans in each ring, denoted as N_s, is determined by the custom angular resolution. Different colors represent different heights.



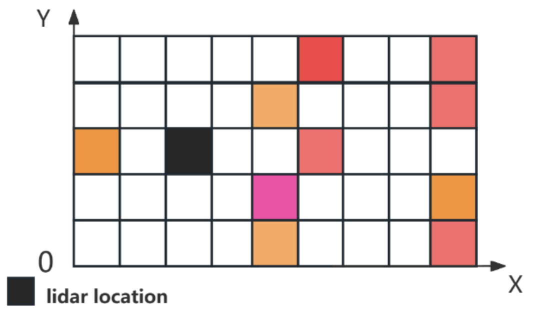
(a) point cloud pose grid map at time m



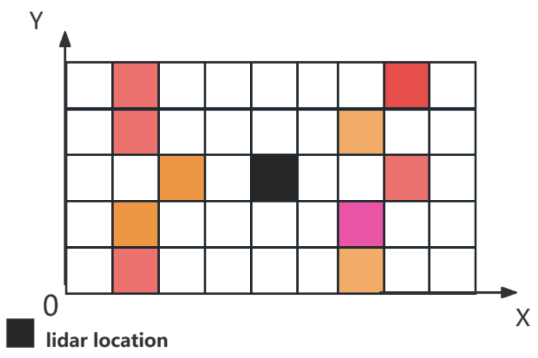
(b) point cloud pose grid map at time n

Fig. 1 2D point cloud pose grid map

Then we construct a point cloud displacement grid map from the three-dimensional point sets $\boldsymbol{H}$ and $\boldsymbol{E}$. This grid map is used to estimate the initial value of the displacement. The point cloud displacement grid map is converted from a scan of point cloud data in the robot coordinate system to a grid point cloud map with the bottom left corner point as the origin. The value of each grid is determined by the highest point cloud height within that grid in Fig. 2.



(a) point cloud displacement grid map at time m



(b)point cloud displacement grid map at time n

Fig. 2 2D point cloud displacement grid map

Due to the limitation of grid map resolution, if the velocity change is gentle, the initial value cannot be corrected by this method. Therefore, the resolution of grid map can be set higher if mobile robot is a low-speed model. Subsequently, the initial pose value is obtained by comparing the point cloud with the grid map. Finally, this initial pose estimation is sent to the ICP (Iterative Closest Point) scan-to-map matching algorithm. Displacement will cause different ranges of point clouds so we need to extract the common view range of point cloud. It is good for matching different kinds of grid maps.

Note that the point cloud orientation grid map and point cloud displacement grid map at a certain historical moment in the robot coordinate system are denoted as $\boldsymbol{J}^B_m$, $\boldsymbol{K}^B_m$ and the other is $\boldsymbol{J}^B_n$, $\boldsymbol{K}^B_n$ at the current time. Then the poses in the world coordinate system corresponding to the two moments are $\boldsymbol{T}^W_m$、$\boldsymbol{T}^W_n$, and the relative pose transformation from m to n is (18).

$$\boldsymbol{T}^m_n = \boldsymbol{T}^{W^-}_m \cdot \boldsymbol{T}^W_n \tag{18}$$

Convert the point cloud in the robot coordinate system obtained at time n to the robot coordinate system at time m:

$$\boldsymbol{P}^B_{n'} = \boldsymbol{T}^{m^-}_n \cdot \boldsymbol{P}^B_n \tag{19}$$

Then the point cloud feature maps $\bar{\boldsymbol{J}}^B_m$、$\bar{\boldsymbol{K}}^B_m$、$\bar{\boldsymbol{J}}^B_n$、$\bar{\boldsymbol{K}}^B_n$ in the same coordinate system are obtained.

Filter $\boldsymbol{P}^B_m$、$\boldsymbol{P}^B_{n'}$according to viewpoint visibility [8], and we can get the common-view point cloud grid map $\tilde{\boldsymbol{J}}^{mn'}_m$、$\tilde{\boldsymbol{K}}^{mn'}_m$、$\tilde{\boldsymbol{J}}^{mn'}_n$、$\tilde{\boldsymbol{K}}^{mn'}_n$.

C. *Initial Position and Orientation Estimation Optimization*

According to the conclusions of M. Brossard [9], the variance of the scan matching algorithm depends on the accuracy of the initial pose estimation. In other words, when the initial position and orientation is accurate enough, the scan matching algorithm achieves higher accuracy.

The initial value optimization is divided into initial position estimation and initial orientation estimation. In initial value optimization, the sequence of initial value estimation needs to be determined. Assuming that the i^{th} scan and the j^{th} scan are two adjacent frames, we obtain the relative pose transformation $\Delta\boldsymbol{q}^{j-1}_j$ from the previous scan to the current scan using the constant velocity motion assumption, with the rotation angle denoted as θ^{j-1}_j. ϑ is the rotation angle threshold. If $\theta^{j-1}_j > \vartheta$, the priority is given to orientation estimation. Otherwise, position estimation takes priority.

The point cloud pose grid maps $\tilde{\boldsymbol{J}}^{mn'}_m$ and $\tilde{\boldsymbol{J}}^{mn'}_n$ are used for orientation estimation. These grid maps are stored as matrices, both having the size of a row and b columns. Let $\boldsymbol{M}^j$ represent the j^{th} column of $\tilde{\boldsymbol{J}}^{mn'}_n$, and N^j denote the j^{th} column of $\tilde{\boldsymbol{J}}^{mn'}_n$. $\boldsymbol{M}_{\Delta q}$ represents the new matrix obtained by shifting all column vectors of $\tilde{\boldsymbol{J}}^{mn'}_m$ to the right by $\boldsymbol{\Delta q}$ columns. For vectors extending beyond column $\mathbf{b}$, they are appended at the beginning of the matrix.

$$\min_{\Delta q} \sum_{j=0}^{b} \left\| \boldsymbol{M}^j_{\Delta q} - \boldsymbol{M}^j \right\|_0 \tag{20}$$

In other words, the purpose of expression (20) is to match Fig. 1 (a) and Fig. 1 (b) and get the $\boldsymbol{\Delta q}$ from time m to n .Obviously, Fig. 1 (a) needs to be translated 3 units in the negative direction of N_s to be consistent with Fig. 1 (b) so $\boldsymbol{q}$ is 3 units which means rotating around the Z-axis of the local coordinate system at time m.

Typically Vehicles do not turn quickly so an exhaustive method can be used to find the optimal value of $\Delta \boldsymbol{q}$. Taking ±10 times of the resolution as the exhaustive interval, the optimal angle of rotation around the Z-axis is expressed as follows:

$$\boldsymbol{q}_{n_0}^{w} = \boldsymbol{q}_{m}^{w} * \boldsymbol{\Delta q} \tag{21}$$

$\boldsymbol{q}_{m}^{w}$ is the global orientation at time m and is derived from the uniform motion model. $\boldsymbol{q}_{n_0}^{w}$ is the initial global orientation at time n_0.

Initial position estimation uses the point cloud displacement grid maps $\widetilde{\boldsymbol{K}}_{m}^{mn'}$ and $\widetilde{\boldsymbol{K}}_{n}^{mn'}$. Firstly, $\widetilde{\boldsymbol{K}}_{m,\Delta q}^{mn'}$ is rotated by $\Delta \boldsymbol{q}$ from time m to n and stored as matrices with dimensions of u rows and v columns. Similar to the process of estimating orientation, the optimal value of Δx and Δy can be solved by an exhaustive search and the difference between $\widetilde{\boldsymbol{K}}_{m,\Delta q}^{mn'}$ transformed by displacement and $\widetilde{\boldsymbol{K}}_{n}^{mn'}$ matrix is computed to obtain the zero norm, which is expressed as (22).

$$\min_{\Delta x, \Delta y} \left\| \widetilde{\boldsymbol{K}}_{m,\Delta q}^{mn'} - \widetilde{\boldsymbol{K}}_{n}^{mn'} \right\|_0 \tag{22}$$

This process helps to get the displacement Δx and Δy by minimizing the zero norm. The displacement correction vector Δt is formed by combining $(\Delta x, \Delta y, 0)$ and initial position at time n can be represented as follows:

$$\boldsymbol{t}_{n_0}^{w} = \boldsymbol{t}_{m}^{w} + \boldsymbol{q}_{n_0}^{w} \otimes \boldsymbol{\Delta t} \tag{23}$$

Where $\boldsymbol{t}_{m}^{w}$ represents the global position at time m . $\otimes$ denotes the operation satisfying quaternion multiplication. $\boldsymbol{q}_{n_0}^{w} \otimes \boldsymbol{\Delta t}$ is the relative displacement from time m to time n. To put it another way, the purpose of expression (22) is to match Fig. 2 (a) and Fig. 2 (b) and get the Δx and Δy on XY direction from time m to n so Δx is 2 units and Δy is 0 units. At last, the vector (2,0,0) is modified to $\boldsymbol{t}_{m}^{w}$.

To sum up, the initial estimates of the orientation $\boldsymbol{q}_{n_0}^{w}$ and the position $\boldsymbol{t}_{n_0}^{w}$ are obtained at the current time n.

III. Experiment

The SLAM system runs on ubuntu 18.04 with AMD 5600H.Based on the open-source work F-LOAM, we design an initial value optimization module and test it on the KITTI dataset.

Firstly, F-LOAM and ours are tested on KITTI under low iteration and their trajectories are shown in Fig. 3 and Fig. 4.

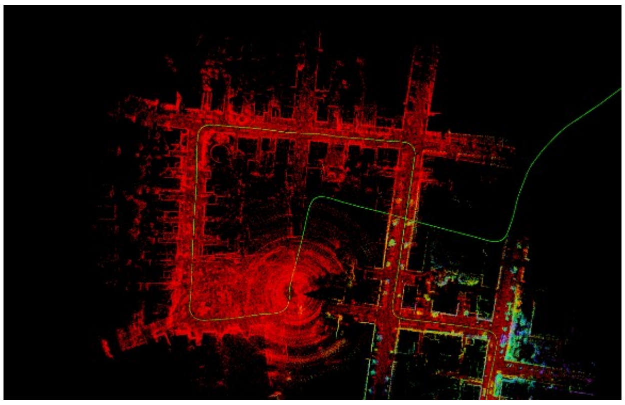


Fig. 3 Trajectory under low iteration. (It obviously shows the trajectory built failed under low iteration count.)

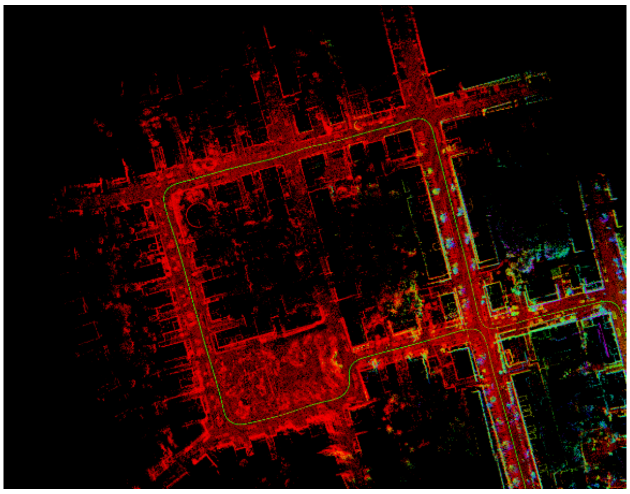


Fig. 4 Trajectory after initial value optimization. (After initial value optimization, the trajectory is built better.)

Because of road containing two continuous turns at 90 degree, the initial value provided by uniform motion model is not enough accurate for ICP and it can’t accurately estimate the pose under low iteration as depicted in Fig. 3.After initial value optimization module, the trajectory has been significantly improved at the same low iterations as depicted in Fig. 4.

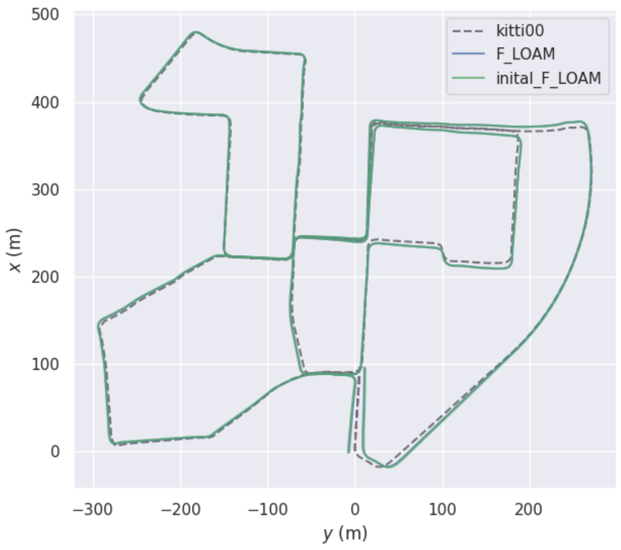


Fig. 5 Trajectory comparison on kitti00

In Fig. 5, the dashed line is ground truth of kitti00. Compared to the blue line generated by F-LOAM, the green line after initial value optimization is closer than it. However, The trajectory after initial value optimization still can not be closed loop.

Table I Evaluation of trajectory accuracy

	max	mean	RMSE	min
Floam	11.33832	3.817776	4.611978	**0.194833**
Ours	**7.040827**	**2.893429**	**3.423987**	0.207286

Table II Evaluation of trajectory smoothness

	median	SSE	STD
Floam	3.273746	96588.609088	2.587455
Ours	**2.414997**	**53237.275980**	**1.830781**

Besides, Table I and Table II show that our method is better than F-LOAM on the max, mean, RMSE(root mean square error), SSE(the sum of squares due to error) and STD(standard deviation) of trajectory error. Max, mean, RMSE, min straightly reflect the error between a trajectory and the ground truth. And the key of analysis is RMSE and STD. RMSE shows the square root of difference between the observed value and the ground truth, which reflects the accuracy of lidar odometer. STD stands for smoothness of difference between a trajectory and the ground truth. Usually, smaller RMSE and STD means a smoothness more accuracy and smoothness trajectory.

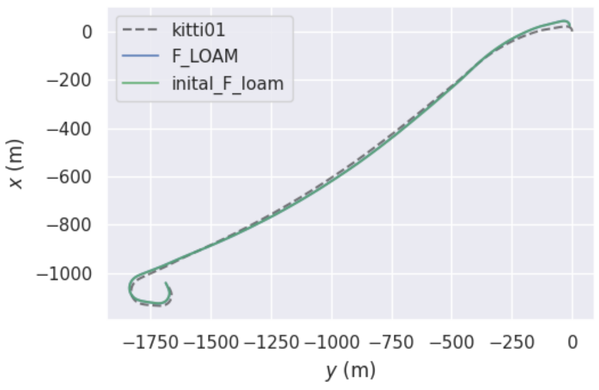


Fig. 6 Trajectory comparison on kitti01

Compared with kitti00, kitti01 has simple environment features and no loop closure. In Fig. 6, the green line and blue line almost overlap so the trajectories can not show which one is better.

Table III Evaluation of trajectory accuracy

	max	mean	RMSE	min
Floam	47.77344	16.10824	18.79549	**0.360834**
Ours	**46.74350**	**16.10200**	**18.74953**	0.399953

Table IV Evaluation of trajectory smoothness

	median	SSE	STD
Floam	16.27044	326775.3196	9.684791
Ours	**16.19694**	**315179.2152**	**9.605754**

However, Table III and Table IV show that trajectory accuracy and smoothness are improved by initial value optimization except for the minimum, which proves the stability of the model of initial value optimization under simple environmental characteristics.

Through the comparative experiment with F-LOAM, this initial value optimization based on the height feature map can solve the question caused by uniform motion model, which makes better location under low iteration and variable speed.

IV. Conclusion and Outlook

In this paper, a solution to uniform motion model which can not accurately estimate initial value at variable speed is proposed, which builds two kinds of grids by extracting height feature and get the initial value by minimizing zero norm. In addition, this method makes the front-end error smaller and has better accuracy at low iterations, which alleviates the error accumulation in the front-end and optimizes the initial value when the uniform speed model is inaccurate and makes lidar odometry more stable. In the future, research should consider the potential effects of resolution of grid map more deeply, which will reduce affect of resolution on accuracy of initial value.

References

[1] W. Hess, D. Kohler, H. Rapp and D. Andor, "Real-time loop closure in 2D LIDAR SLAM," 2016 IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, 2016, pp. 1271-127.

[2] Zhang J, Singh S . LOAM: Lidar Odometry and Mapping in Real-time[C]//Robotics: Science and Systems Conference.2014.

[3] H. Wang, C. Wang, C. -L. Chen and L. Xie, "F-LOAM: Fast LiDAR Odometry and Mapping," 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Prague, Czech Republic, 2021, pp. 4390-4396.

[4] T. Shan and B. Englot, "LeGO-LOAM: Lightweight and Ground-Optimized Lidar Odometry and Mapping on Variable Terrain," 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 2018, pp. 4758-476.

[5] L. Liao, C. Fu, B. Feng and T. Su, "Optimized SC-F-LOAM: Optimized Fast LiDAR Odometry and Mapping Using Scan Context," 2022 6th CAA International Conference on Vehicular Control and Intelligence (CVCI), Nanjing, China, 2022, pp. 1-6.

[6] G. Kim and A. Kim, "Scan Context: Egocentric Spatial Descriptor for Place Recognition Within 3D Point Cloud Map," 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 2018, pp. 4802-4809.

[7] P. J. Besl and N. D. McKay, "A method for registration of 3-D shapes,"in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 14, no. 2, pp. 239-256, Feb. 1992.

[8] D. Yoon, T. Tang, and T. Barfoot, "Mapless online detection of dynamic objects in 3D LiDAR," in Proc. Conference on Computer and Robot Vision (CRV), 2019, pp. 113–120.

[9] M. Brossard, S. Bonnabel and A. Barrau, "A New Approach to 3D ICP Covariance Estimation," in IEEE Robotics and Automation Letters, vol. 5, no. 2, pp. 744-751, April 2020.