LIGHTWEIGHT AND FAST MATCHING METHOD FOR LIDAR-INERTIAL ODOMETRY AND MAPPING

Chuanjiang Li,* Ziwei Hu,* Yanfei Zhu,* Xingzhao Ji,* Chongming Zhang,* and Ziming Qi**

Abstract

This paper presents lightweight and fast Lidar-inertial odometry (LF-LIO) for a robot's real-time pose estimation in an unknown complex environment. This system includes prediction, odometry, mapping, and trajectory optimisation modules. In the prediction module, the initial value of the odometer's motion is calculated by inertial measurement unit (IMU) pre-integration and the state of the previous moment, the odometry then employs a scan-to-submap matching method based on ground segmentation and optimisation proposed by this paper to estimate the pose transformation between consecutive frames. To ensure high performance in real-time, a keyframe map is created instead of a full map. When updating incrementally the efficiency of the map is improved, meanwhile an efficient dynamic sliding window is proposed to manage submaps. We compare the performance of LF-LIO with three methods, Lidar odometry and mapping in real-time (LOAM), lightweight and ground-optimised Lidar odometry and mapping on variable terrain (LeGO-LOAM), and fast direct Lidar-inertial odometry (Fast-LIO2), using KITTI datasets, the contrasted results of the application indicate that the proposed LF-LIO method has better accuracy with a reduced computational burden.

Key Words

SLAM, location, keyframe, mapping

1. Introduction

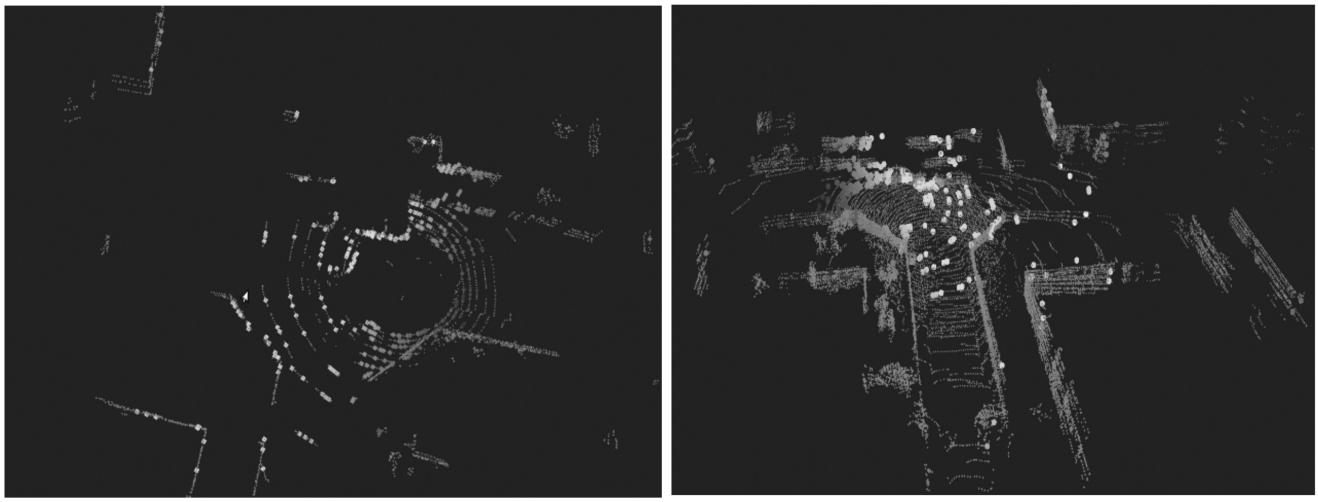
Simultaneous localisation and mapping (SLAM) in an unknown environment are one of the key technologies for a robot's autonomous navigation. The robot's onboard computing resources are tight. The SLAM algorithm deployed on the robot requires fewer computing resources to achieve high-precision, low-latency, and high-robust position estimation. Visual methods [1]–[7] have advantages in obstacle recognition and low-texture scenes and have high positioning accuracy, however, the visual method has a small observation range and is easily affected by illumination changes. To process high-resolution data acquired by visual sensors in real time, additional computing resources are required. Multi-line Lidar is widely used in various unmanned systems because of its wide sensing range, unaffected by ambient light, and the ability to directly obtain the distance between sensors and obstacles in three-dimensional space, *e.g.*, self-driving cars [8], and factory robots [9].

Previous researchers have proposed various methods to apply Lidar to achieve robot motion state estimation and simultaneous mapping. A paper [10] proposed a Lidar odometry and mapping in real-time (LOAM), which is an advanced iterative closest point (ICP) matching method [11]–[14]. A full point cloud is replaced by the plane and edge features extracted from the point cloud to match the pose. This pose estimation has two parts, the front-end uses frame-to-frame feature point matching to output high-frequency and low-precision pose estimates at a frequency of 10 Hz, and uses this estimate to remove motion distortion in the point clouds. The back end forms a small map every 10 frames of point clouds, to match the feature points with the submap extracted from the complete map. Outputting low-frequency and high-precision pose estimates at a frequency of 1 Hz, to complete a real-time, low-drift pose estimation, which is the most accurate of a single Lidar odometer in the KITTI [15]. Many subsequent works are based on the framework proposed in LOAM [10], *e.g.*, a lightweight and ground-optimised Lidar odometry and mapping on variable terrain (LeGO-LOAM) [16]. Based on LOAM, applying fast segmentation of 3D point clouds [17], the ground points are segmented out to constrain the altitude error of the odometry estimate. Based on LOAM, a Lidar odometry, and mapping with loop-closure detection-based correction (LLOAM) [18] added loopback detection to eliminate accumulated errors. In a fast, robust, high-precision Lidar odometry and mapping package for Lidars of small fov (Loam-livox) [19] the LOAM method is adapted according to the characteristics of solid-state Lidar.

* College of Information, Mechanical and Electrical Engineering, Shanghai Normal University, Shanghai, China; e-mail: yfzhu@shnu.edu.cn

** New Zealand Maritime School, Manukau Institute of Technology, Auckland, New Zealand
Corresponding author: Yanfei Zhu

Recommended by Jiankun Wang
(DOI: 10.2316/J.2024.206-0880)



(a) (b)

Figure 1. (a) Corresponding feature points were detected from frame-to-frame and (b) corresponding feature points were detected from frame-to-map.

At the data acquisition unit, sensors, such as IMU and global navigation satellite system (GNSS), *etc.*, provide initial values for laser odometry matching, eliminate motion distortion and accumulated errors, to improve odometer accuracy and robustness. A fast direct Lidar-inertial odometry (Fast-LIO2) [20] employs raw point cloud frames for matching to effectively use subtle features in the environment and uses extended Kalman filtering (EKF) to fuse IMU data, to enable fast, low-drift odometers. A 3D LiDAR SLAM integration with GPS/INS for UAVs in urban GPS-degraded environments [21] introduces GNSS data as priors during optimisation to eliminate the accumulated error of the odometer. A tightly-coupled Lidar inertial odometry *via* smoothing and mapping (LIO-SAM) [22] uses factor graphs to jointly use the Lidar odometry and IMU pre-integration data [23] as pose constraints, and joint optimisation. When GNSS is available, the GNSS factor is introduced as the prior value of pose to eliminate the accumulated error of the odometer. Tightly-coupled Lidar-visual-inertial odometry *via* smoothing and mapping (LVI-SAM) [24] uses visual odometry, Lidar odometer, and IMU pre-integration data as constraints for pose optimisation, and it uses vision to identify obstacles, advantages in loopback detection, to build a higher accuracy odometer.

With limited computing resources, and the onboard robotic systems, it is also still challenging to estimate the state of a moving robot and build a map synchronously, employing multi-line LiDAR to enable fast, low drift robot state estimation and simultaneous map construction. Research questions are highlighted:

1) Though the current resolution of multi-line Lidar is gradually improved, hundreds of thousands of points of measurement data are generated per second to process the data in real-time to estimate the pose and movement of the robot, it requires more computationally efficient methods.
2) The farther distant the multi-line Lidar scans, the lower the resolution can be achieved, therefore, commonly using frame-to-frame feature point matching methods, farther distant feature points cannot be effectively completed. Such an odometer has relatively low accuracy. As shown in Fig. 1, compared to frame-to-frame matching, frame-to-map matching can discover more distant feature points.
3) The robot needs to maintain a complete map during the operation of the algorithm. As the runtime increases, the memory usage of the system increases, and real-time performance reduces. It was extremely inefficient using loop closure constraints to correct accumulated errors of the odometer, and also need to remap the whole map.

This work proposes a lightweight and fast Lidar-inertial odometry (LF-LIO) and mapping method to solve the above problems. Firstly, the raw IMU data is processed by using a pre-integration model. The predicted motion during one lidar scan is used to compensate for motion distortion in the Lidar DAT and also provides an initial value for the odometer matching. Odometry takes segmentation clustering to remove useless points from the point cloud, to match the feature points extracted from the point cloud with the submap, and to better use of environmental features. The map is composed of keyframes [25] containing the feature point cloud and the corresponding estimated pose. Adding new keyframes to the map takes less time compared to maintaining a full map, and the map does not need to be remapped after global pose optimisation. The system also employs graph optimisation to fuse absolute GNSS measurements and detected loop closure constraints to eliminate accumulated errors, to improve the accuracy of pose estimation. This paper has the following novel research outcomes:

1) By pre-integrating the IMU and combining the robot state, a highly reliable matching initial value is calculated to improve the convergence speed of point cloud matching.

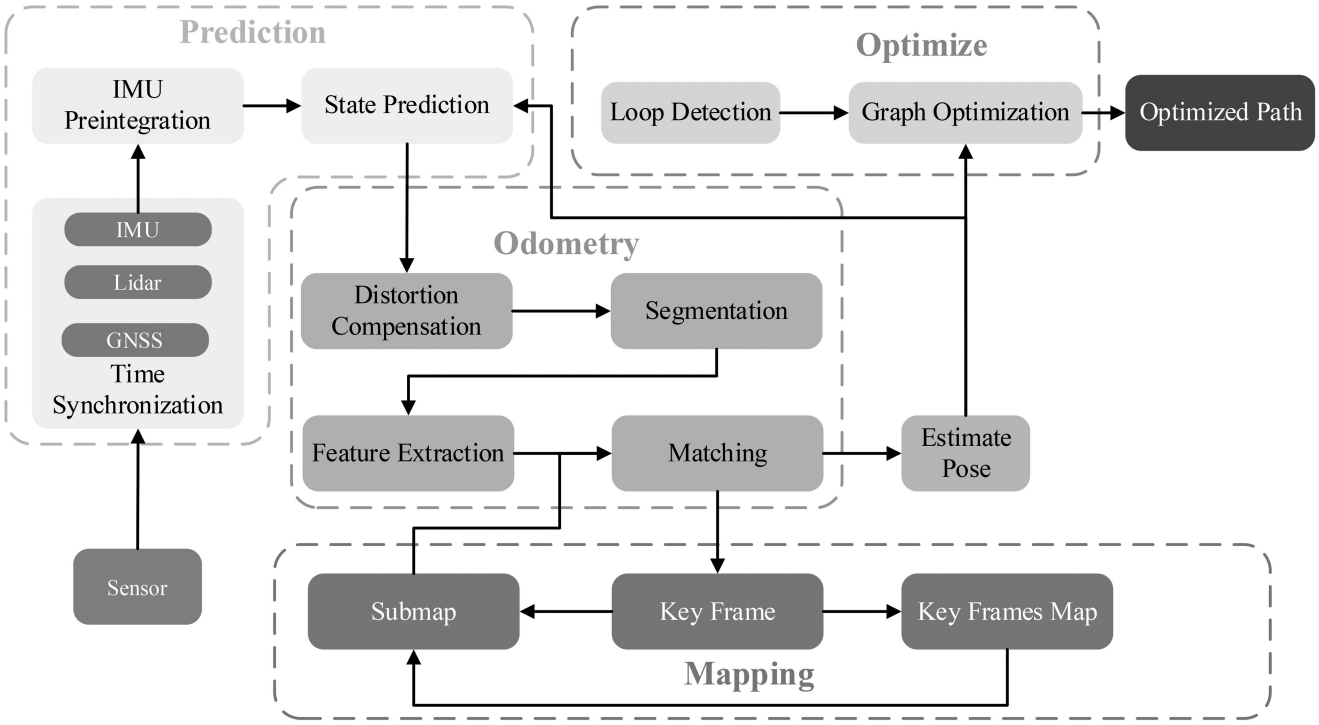


Figure 2. System overview.

2) A lightweight and fast frame-to-map odometer matching method based on point cloud segmentation clustering and dynamic sliding window is proposed. By reducing useless feature points, the feature points that are far away from the radar in the point cloud are better utilised to improve the accuracy and matching speed of the odometer.
3) A new map form is proposed. The map consists of a series of keyframes. Each keyframe contains a point cloud composed of edge points, a point cloud that composed of plane points, and the estimated value of the robot state corresponding to the key frame.

1.1 Notation

This paper defines the following symbols: W represents the world coordinate system. B represents the robot body coordinate system. Assuming that the IMU and RTK coordinate systems coincide with the robot body coordinate system, the state of the robot is defined as:

$$X = \left[R^T, p^T, v^T\right]^T \tag{1}$$

While $R \in \mathrm{SO}(3)$ represents a rotation matrix, $p \in R^3$ indicates the displacement of the robot, v represents the speed of the robot. $T_{BW} \in \mathrm{SE}(3)$ represents the pose transformation from the world coordinate system to the robot body coordinate system $T_{BW} = [R_{BW}|t_{BW}]$.

2. Lightweight Lidar - Inertial Odometry and Mapping

2.1 System Overview

As shown in Fig. 2, the system structure proposed is divided into four main modules, motion prediction, odometer estimation, trajectory optimization, and mapping.

The prediction module time-synchronises all sensor data received to ensure data consistency in time, then, pre-integrate the IMU data to get the relative delta between two frame states, to combine with the optimal estimated value of the system speed state at the start of pre-integration, to calculate the pose prediction value. The odometer obtains the motion prediction value, to compensate for motion distortion in point clouds based on predicted values, to segment the point cloud and extract features. The extracted feature is matching the points and sub-maps, to get the relative pose transformation of the point cloud of the current frame relative to the previous frame, and calculate the optimal estimate of the current state of the system. Finally, the odometer outputs the optimal pose estimation at the current moment and the feature points extracted from the point cloud. The mapping module saves the odometer outputs that meet the same conditions as new keyframes, to be added to sub-maps and environment maps. The trajectory optimisation module uses graph optimisation, to combine with GNSS and detected loop closure constraints, to perform global pose optimisation, eliminate the accumulated error of the odometer, to improve the global consistency of pose estimation.

The basic formula and workflow of each module are described as below.

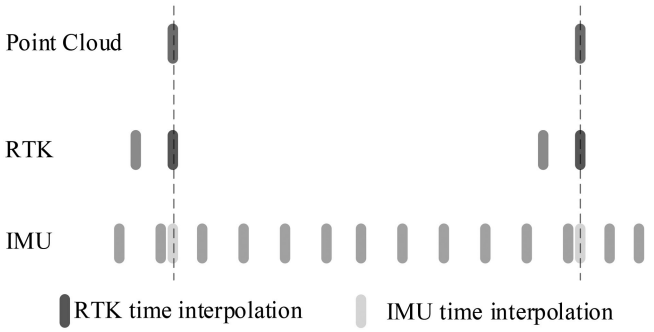


Figure 3. Sensor data interpolation.

2.2 Prediction

It is important that multi-sensor data clock is synchronised. If the measurement data at different times is used for state prediction, it will interfere with each other and reduce the accuracy of the result. Once the IMU and GNSS data is acquired and interpolated, the equivalent value in the laser point cloud is calculated based on (2):

$$\mathrm{D} = \frac{t_{\mathrm{back}}-\mathrm{t}}{t_{\mathrm{front}}-t_{\mathrm{back}}} D_{\mathrm{front}} + \frac{\mathrm{t}-t_{\mathrm{front}}}{t_{\mathrm{front}}-t_{\mathrm{back}}} D_{\mathrm{back}} \tag{2}$$

t indicates the acquisition time of the current point cloud frame, t_{front} and t_{back} represents the acquisition time of the two frames of data closest to time t. D_{front} and D_{back} represent the corresponding data. The schematic diagram of interpolation is shown in Fig. 3.

After clock synchronisation, the motion state of the robot is predicted based on the measurement data of the IMU. The measured values of the angular velocity and linear acceleration of the robot measured by the IMU can be modelled as:

$$\widetilde{\omega_t} = \omega_t + b_t^{\omega} + \eta_t^{\omega} \tag{3}$$

$$\widetilde{a_t} = R_t^{\mathrm{BW}} (a_t - \mathrm{g}) + b_t^{a} + \eta_t^{a} \tag{4}$$

While $\widetilde{\omega_t}$ and $\widetilde{a_t}$ indicate the original measurement value of the IMU under B at time t, the measured value consists of the true value, zero bias b_t and white noise η_t. In this article, we assume that η_t is zero, and therefore, it is not mentioned in the subsequent equations. R_t^{BW} is the transition matrix from W to B at time t, g is the gravitational acceleration at W. The state of the ith frame is known, the motion state of the robot at the jth frame can be expressed as:

$$R_j = R_i \prod_{k=i}^{\mathrm{j}-1} \mathrm{Exp}\left(\left(\widetilde{\omega_k} - b_k^{\omega}\right) \Delta t_{k,k+1}\right) \tag{5}$$

$$v_j = v_i + \mathrm{g}\Delta t_{ij} + \sum_{k=i}^{\mathrm{j}-1} R_k \left(\widetilde{a_k} - b_k^{a}\right) \Delta t_{k,k+1} \tag{6}$$

$$p_j = p_i + \sum_{k=i}^{\mathrm{j}-1} \left(v_k \Delta t_{k,k+1} + \frac{1}{2}\mathrm{g}\Delta t_{ij}^2 \Delta t_{k,k+1}^2 + \frac{1}{2} R_k \left(\widetilde{a_k} - b_k^{a}\right) \Delta t_{k,k+1}^2 \right) \tag{7}$$

To avoid the cumulative error caused by the introduction of IMU direct integration, to improve forecast accuracy, the relative motion between two adjacent frames is calculated using the IMU pre-integration model proposed in on-manifold pre-integration for real-time visual–inertial odometry [23], is defined as:

$$\Delta R_{ij} = R_i^T R_j \tag{8}$$

$$\Delta v_{ij} = \sum_{k=i}^{\mathrm{j}-1} \Delta R_{ik} \left(\widetilde{a_k} - b_k^{a}\right) \Delta t_{k,k+1} \tag{9}$$

$$\Delta p_{ij} = \sum_{k=i}^{\mathrm{j}-1} \left[\Delta v_{ik} \Delta t_{k,k+1} + \frac{1}{2} \Delta R_{ik} \left(\widetilde{a_k} - b_k^{a}\right) \Delta t_{k,k+1}^2 \right] \tag{10}$$

Then, according to the pre-integration results and the optimal estimation of the robot state at the previous moment $\widetilde{X}_{t-1}$, from (11)–(13):

$$R_t = R_{t-1} \Delta R_{t-1,t} \tag{11}$$

$$v_t = v_{t-1} + R_t \Delta v_{t-1,t} + \mathrm{g}\Delta t_{t-1,t} \tag{12}$$

$$p_t = p_{t-1} + R_{t-1} \Delta p_{t-1,t} + v_{t-1} \Delta t_{t-1,t} + \frac{1}{2} g \Delta t_{t-1,\mathrm{t}}^2 \tag{13}$$

Calculate the predicted value of the system state at the current moment X_t, and the predicted value of relative motion from the previous frame $\Delta T_{t-1,t} = [\Delta R_{t-1,t} | \Delta t_{t-1,t}]$.

Remark 1: The speed at the last moment used in the calculation is calculated based on the pose matched by the Lidar, it can be considered as the true value of the real-time speed of the robot, the pre-integration value only represents the pose change increment between two adjacent frames regardless of the state at any other time, meanwhile, the measurement accuracy of the IMU is very high in a short time. Thus the calculated predictions are highly reliable.

2.3 Odometry

2.3.1 Distortion Compensation

Whenever a new point cloud frame $\widetilde{F}_i$ is received, point clouds are motion distortion compensated based on the scanning principle of mechanical radar, and the relative motion state prediction value given by the prediction module $\Delta T_{i-1,i}$. Using (14) to calculate the pose transformation of P_k^i relative to $\widetilde{F}_i$ at starting time T_k^i:

$$T_k^i = \frac{t_k}{\Delta \mathrm{t}} \Delta T_{i-1,i} \tag{14}$$

Where t_k is the difference between the acquisition time of each point P_k^i in $\widetilde{F}_i$ and the start time of $\widetilde{F}_i$. Δt is the time spent scanning the current point cloud frame. Note, (14) is merely a conceptual formula, pose transformation matrix can't directly calculate interpolation results by multiplication. P_k^i is projected to $\widetilde{F}_i$ in the coordinate system B_i at the beginning, thereby, the motion distortion

Algorithm 1: Lidar Inertial Odometry

Input: Lidar frame $\widetilde{F}_i$

IMU measurement ($\widetilde{\omega}_t$, $\widetilde{a}_t$ during current frame)

Last optimal state estimation $\widetilde{X}_{i-1}$ and submap $M_{i-1}\}$

1 Throw $\widetilde{\omega}_t$, $\widetilde{a}_t$ and (8)- (13) to prediction X_i and $\Delta X_{i-1,i}$

2 Compensate distortion for $\widetilde{F}_i$ with $\Delta X_{i-1,i}$ and (2).

3 Segment $\widetilde{F}_i$ to $F_i^{e'}$ and $F_i^{p'}$

4 **for** *each edge point in* $F_i^{e'}$ do

5 Find an edge line in M_{i-1} as the correspondence, compute point to line distance based on (14) and stack the equation to (16)

6 **for** *each planar point in* $F_i^{p'}$ **do**

7 Find a planar patch in M_{i-1} as the correspondence, compute point to plane distance based on (15) and stack the equation to (16)

8 solve (16) by Gauss Newton, calculate the $\widetilde{T}_i^{\mathrm{BW}}$

9 Update $\widetilde{X}_i$ by $\widetilde{T}_i^{\mathrm{BW}}$

Output: Feature $F_i^{'}$, state estimation $\widetilde{X}_i$

caused by the radar motion of the point cloud data is removed.

2.3.2 Split Clustering

Undistorted point cloud is split clustered, to ground and obstacle points in point cloud, to cluster obstacle points, and classify points from the same obstacle [16]. After split clustering, obstacles that contain too few scan points are removed, to avoid extracting unreliable features during feature extraction.

Remark 2*:* When clustering obstacles, Lidar has smaller vertical resolution, the method in fast range image-based segmentation of sparse 3D laser scans for online operation [26] is not suitable when clustering two points that are vertically adjacent, Euclidean distance is applied to determine whether two vertically adjacent points come from the same obstacle.

2.3.3 Feature Extraction

To extract point cloud features after segmentation and clustering, the method of feature extraction is applied with similar to the method proposed in lightweight and ground optimised Lidar odometry and mapping on variable terrain [16]. As it is confirmed that the planar features in obstacles help improve the matching accuracy of heading angle and horizontal movement, the extraction range of plane features has been extended. In addition to extracting planar features at ground points, planar feature extraction is also performed at non-ground points. $\widetilde{F}_i$ is extracted of edge features and plane feature, $F_i^{e'}$ and $F_i^{p'}$, which construct feature set of $\widetilde{F}_i$, $F_i^{'} = \{F_i^{e'}, F_i^{p'}\}$.

2.3.4 Scan Match

Matching with $F_i^{'}$ and submap M_{i-1}, pose state $F_i^{'}$ can be calculated. For computational efficiency and robustness in complex outdoor environments, the method proposed in LOAM is chosen. Robot pose T_{i-1}^{BW} can be predicted by IMU, to project $F_i^{'}$ from B to W to get F_i^{w}, according to the feature points in F_i^{w}, to find five points closest to the Euclidean distance of the feature point in M_{i-1} to calculate the covariance matrix of these five points, and find their three eigenvalues. If one of them is significantly larger than the other two, it can be considered that the line feature corresponding to the edge feature point has been found, if two eigenvalues are significantly larger than the other, it can be considered that the plane feature corresponding to the plane point is found. Use the Euclidean distance between the feature points and the corresponding features in the submap to construct the error equation. Euclidean distance calculation formula for edge point and line feature and plane point and plane feature as (15) and (16):

$$d_k^e = \frac{\left|\left(F_{i,k}^e - M_{i-1,u}^e\right) \times \left(F_{i,k}^e - M_{i-1,v}^e\right)\right|}{\left|M_{i-1,u}^e \times M_{i-1,v}^e\right|} \tag{15}$$

$$d_k^p = \frac{\begin{vmatrix} \left(F_{i,k}^p - M_{i-1,u}^p\right) \\ \left(M_{i-1,u}^p - M_{i-1,v}^p\right) \times \left(M_{i-1,u}^p - M_{i-1,w}^p\right) \end{vmatrix}}{\left|\left(M_{i-1,u}^p - M_{i-1,v}^p\right) \times \left(M_{i-1,u}^p - M_{i-1,w}^p\right)\right|} \tag{16}$$

k, u, v, w represents the feature point number. For edge points $F_{i,k}^e$, $M_{i-1,u}^e$, and $M_{i-1,v}^e$ are the two points on their corresponding line features. For plane points $F_{i,k}^p$, $M_{i-1,u}^p$, $M_{i-1,v}^p$, and $M_{i-1,w}^p$ are the three points that make up its plane feature, the (15) and (16) are solving a

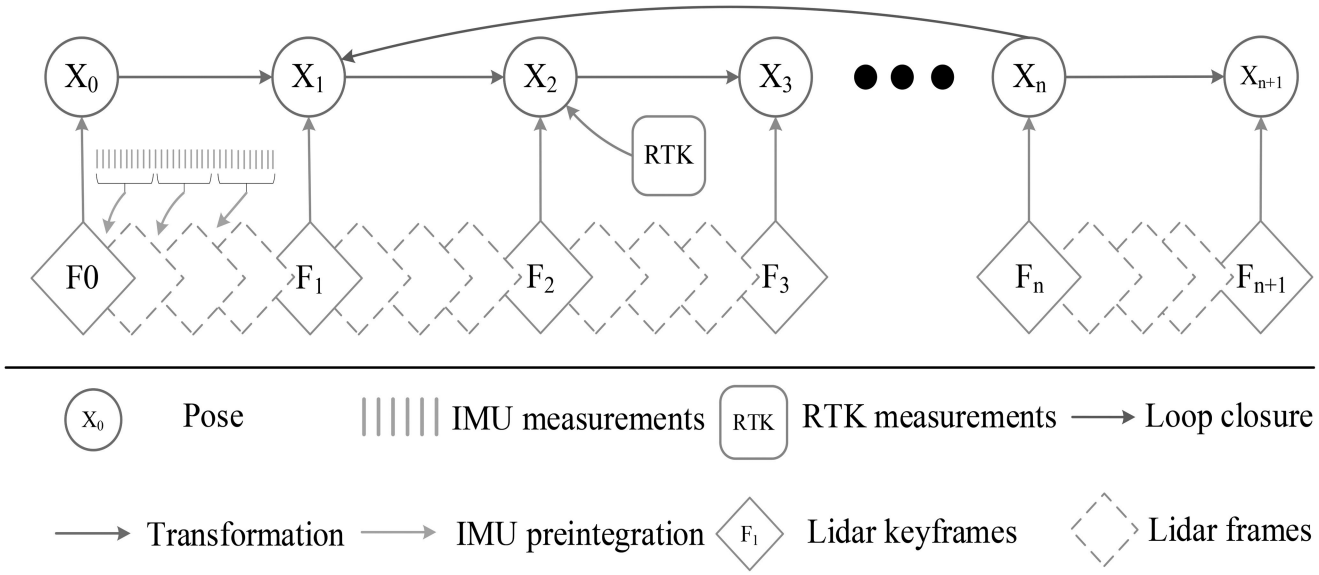


Figure 4. Pose graph structure.

Table 1
Average Processing Time for a New Scan

Matching method	LF-LIO	LOAM	LeGO-LOAM	Fast-LIO2
Scan To scan	/	38.43 ms	12.76 ms	35.33 ms
Scan To map	17.98 ms	/	/	/
Map To map	/	225.4 ms	67.79 ms	/

least squares problem instead of solving the optimal pose estimation:

$$\min_{BW_{\widetilde{T}_i}} \left\{ \sum_{F^e_{i,k} \in F^e_i} d^{e2}_k + \sum_{F^p_{i,k} \in F^p_i} d^{p2}_k \right\}$$

Gauss Newton method is used to solves the optimal pose estimation $\widetilde{T}_i$ to make the equation get the minimum value. Then, the relative motion between adjacent frames $\Delta T_{i-1,i}$ can be calculated by (17):

$$\Delta T_{i-1,i} = \widetilde{T}_{i-1}\widetilde{T}_i \tag{17}$$

Finally based on $\widetilde{T}_i^{\mathrm{BW}}$ and $\Delta T_{i-1,i}$, the optimal estimated value of the robot's motion state in the current frame is output. Overall algorithm for odometer is shown in Algorithm 1.

2.4 Mapping

While each frame of the point cloud is inserted into the environment map after matching, the performance of the system will be affected. Therefore, the concept of keyframes commonly used in vision methods is introduced. When the cumulative motion transformation of the robot relative to the previous key frame is greater than the set threshold, a new keyframe is created with one frame containing point cloud features $F_i^{'}$ and the corresponding optimal pose estimation $\widetilde{T}_i^{\mathrm{BW}}$, point cloud frames between two keyframes are discarded after matching.

If the keyframes are stitched into a complete point cloud map, the point cloud map needs to be re-stitched after each trajectory optimisation, it is waste a lot of computing resources. Thus LF-LIO environmental map consists of a series of individual keyframes. After each pose optimisation, the corresponding poses are corrected in the keyframes, no need to be projected to the global map. Meanwhile, this storage method has an advantage, earlier keyframes can save the point cloud into a disk, and only the pose corresponding to the keyframe is saved in memory to reduce memory consumption, to support a larger point cloud map.

The map-based storage format system uses the idea of sliding windows to manage submaps containing a dynamic number of keyframes for matching with point cloud frames. The number of keyframes contained in the submap has a minimum value min and a maximum value max. When a new keyframe is added to the submap, if the number of keyframes in the submap is not greater than max, insert directly into submap, otherwise, extract the most recent min keyframes, to generate new submap $M_i = \{F_i^{'}, F_{i-1}^{'}, \ldots\ldots F_{i-\min}^{'}\}$.

2.5 Trajectory Optimise

To eliminate the accumulated error of the odometer to build high-precision point cloud maps, the system

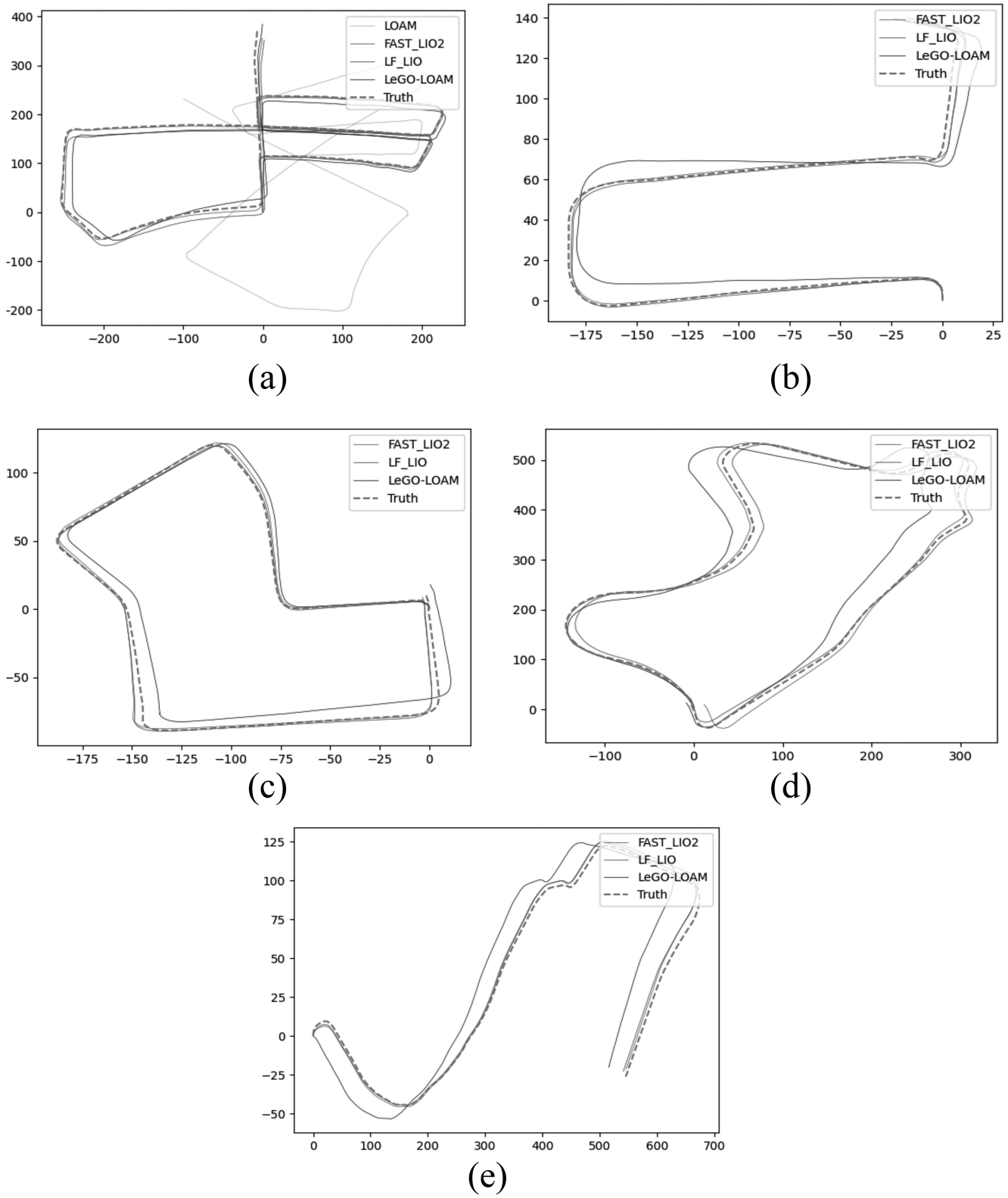


Figure 5. Odometry trajectories of LeGO-LOAM, Fast-LIO2, and LF-LIO in NO. 18, NO. 22, NO. 27, NO. 33, NO. 34 data.

employs graph optimisation to combine absolute GNSS measurements with loop closure constraints detected by the system. The structure of the pose graph is shown in Fig. 4. The keyframe pose is set as the node to be optimised, and constraint edges are built as the pose transformation of adjacent keyframes and detected loops. When the error between the odometer poses and the GNSS absolute measurement value is greater than the set threshold, the corresponding GNSS measurements are taken as a priori, then they are globally optimised to eliminate accumulated errors and improve system accuracy.

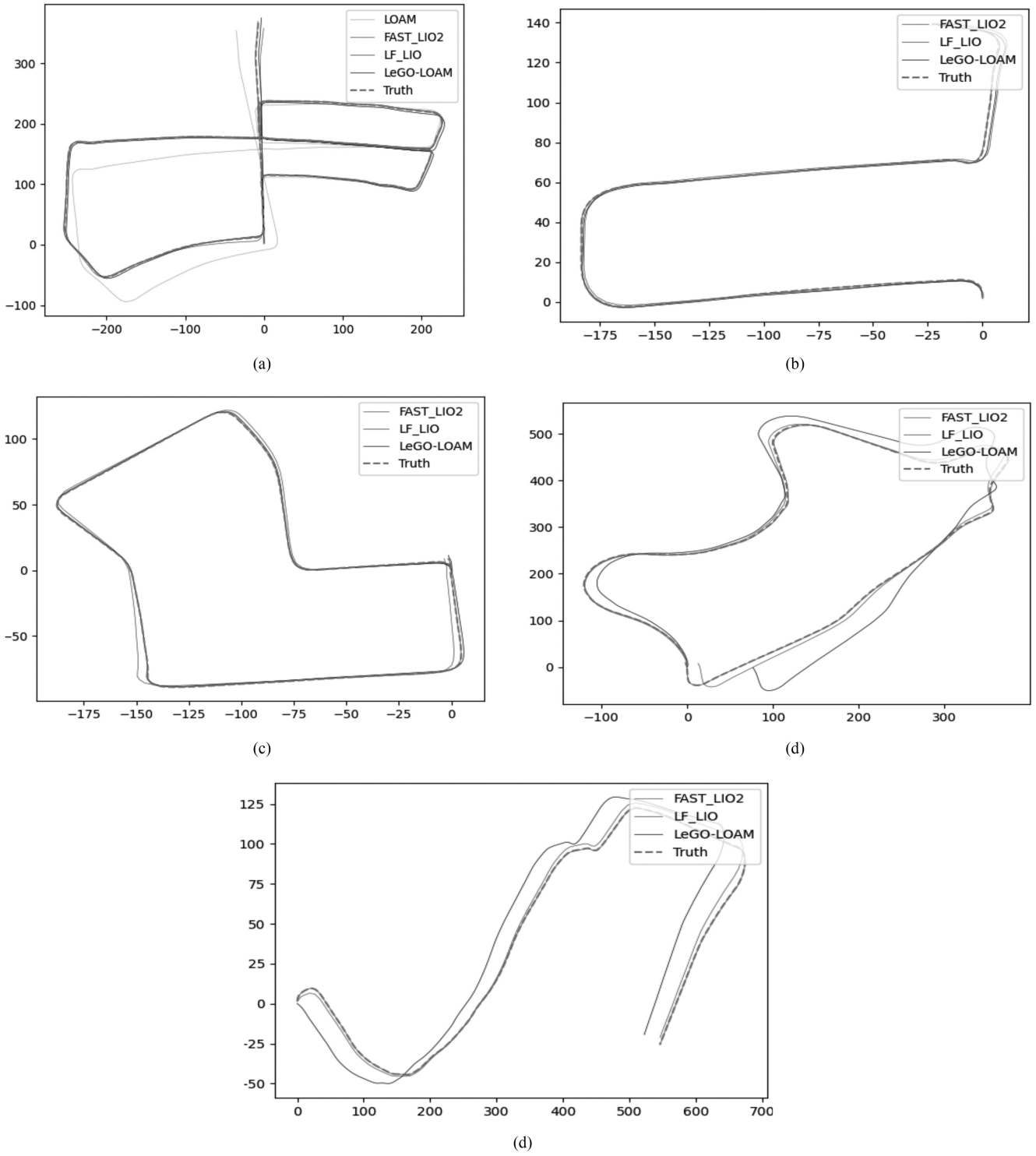
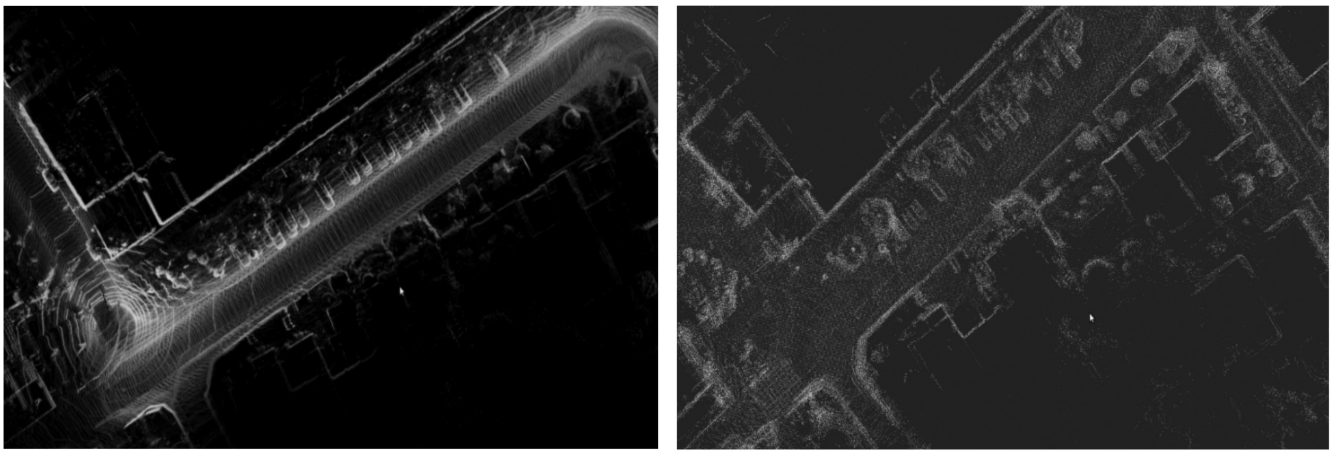


Figure 6. Optimised trajectories of LeGO-LOAM, Fast-LIO2, and LF-LIO in NO. 18, NO. 22, NO. 27, NO. 33, NO. 34 data.

3. Evaluation Test

To verify the performance of the proposed method, this paper conducts a series of tests on the standard KITTI dataset. The KITTI dataset is jointly released by the Karlsruhe Institute of Technology in Germany and the Toyota American Institute of Technology, Dataset includes Velodyne 64-line 3D Lidar, IMU, GPS, and Information captured by the four cameras, covering many scenes, such as villages, cities, highways, *etc.* This article selects the NO. 18, NO. 22, NO. 27, NO. 33, NO. 34 data in KITTI dataset for tests, and compared with LOAM, LeGO-LOAM, and Fast-LIO2, using a laptop with i7-9700HQ as the computing platform, OS is ubuntu 18.04, applying ROS [27] as the communication framework for each module. All four algorithms are implemented in C++.

Remark 3: LOAM is the earliest SLAM method based on multi-layer laser radar. Some methods proposed



(a) (b)

Figure 7. Local point cloud map of Fast-LIO2 and LF-LIO after repeated paths: (a) is Fast-LIO2 and (b) is LF-LIO.

by LOAM are used in almost every multiline laser radar odometer, and the accuracy of LOAM ranks at the top of the laser radar odometer accuracy ranking of the KITTI dataset all year round, but its subsequent work has not been open source. The accuracy of open-source code is low, so we do not specifically compare the accuracy of LOAM. On the basis of LOAM, Lego-LOAM makes some optimisations for feature extraction and pose estimation. Compared with LOAM, Lego-LOAM has a much faster computing speed that can run well on embedded boards with lower computing resources. Many mobile robots with autonomous positioning capabilities currently produced are applied to LeGO-LOAM, which is also an advanced method recognised by researchers. FAST-LIO2 is the latest method just published this year. Because it proposes a new map format with better efficiency, it can spend redundant computing resources on matching. It directly uses the full volume point cloud for matching, so it has high accuracy. It is also a representative method of Kalman filter fusion.

3.1 Odometry Efficiency

To test the efficiency of the LF-LIO proposed match method, comparison with LOAM, LeGO-LOAM, and Fast-LIO2 on the KITTI dataset, average time is taken to test four odometers processing a new frame of point cloud data, and the average time is taken by LOAM and LeGO-LOAM to perform map to map matching. The results are shown in Table 1. The speed of the LF-LIO method to process a frame of the point cloud, only slower than the method proposed by LeGO-LOAM. When the data frame rate of the laser spinner is 10 HZ, the odometer can be run in real-time.

3.2 Odometry Accuracy

The system is also tested with only the front-end odometer for open-loop estimation, LeGO-LOAM, Fast-LIO2, and LF-LIO trajectories as shown in Fig. 5. The root mean square errors of the estimated trajectories and the real trajectories of the three methods are shown in Table 2.

Table 2
The RMSE Transformation Error of the Three Methods Compared to the Real Trajectory

Method/dataset number	LeGO-LOAM	Fast-LIO2	LF-LIO
18	14.38	6.78	**6.04**
22	9.89	2.87	**2.69**
27	10.47	5.49	**4.10**
33	54.48	**17.31**	23.83
34	36.19	20.74	**18.10**

It can be seen that running long enough distances, the deviation of LF-LIO from the real trajectory is smaller than that of LeGO-LOAM and Fast-LIO2 in data NO.18, NO. 22, NO.27, NO. 34. Only data NO.33, Fast-LIO2 has better accuracy.

3.3 Trajectory Optimisation and Mapping

The optimised trajectories of the three methods are shown in Fig. 6.

In the LF-LIO method proposed in this paper, the front end uses scan to map, the odometer is extremely accurate and incorporates GNSS data as prior data, meanwhile, a loopback constraint is introduced to continuously eliminate the accumulated error during the driving process. Therefore, the optimised trajectory is basically the same as the true value trajectory. The root means square errors of the estimated trajectories and the real trajectories of the three methods are shown in Table 3.

Whether there is ghosting in the point cloud map has a great impact on subsequent relocation and path planning. Local maps of repeatedly traversed road sections by zooming in on FAST-LIO2 and LF-LIO, as shown in Fig. 7. It can be seen that the map created by LF-LIO has almost no ghosting in the repeated passing places, and Fast-LIO2 because of the cumulative error when passing through the

Table 3
The Optimized RMSE Transformation Error of the Three Methods Compared to the Real Trajectory

Method/dataset number	LeGO-LOAM	Fast-LIO2	LF-LIO
18	5.36	6.78	**0.45**
22	4.72	2.87	**1.65**
27	4.64	5.49	**1.96**
33	46.06	17.31	**2.17**
34	28.72	20.74	**0.21**

repeated road sections, there will be errors due to the pose used to stitch the point cloud, Causing the resulting map to be ghosted.

4. Conclusion

This paper proposes a lightweight and fast radar-inertial odometry and mapping method based on frame-to-map matching, LF-LIO, which is used for robots to estimate their own position in real-time and build an environment map in a complex outdoor environment.

Odometer uses frame-to-map matching, to better use of feature points extracted from point clouds. Applying keyframe maps instead of full maps avoids system performance degradation caused by the reprojection of large maps after pose optimisation. The proposed method is thoroughly tested and evaluated on the KITTI dataset. Compared with LOAM, LeGO-LOAM, and Fast-LIO2, when the system consumes fewer computing resources, this proposed method got better accuracy. Meanwhile, thanks to the high-precision odometer, the optimised pose consistency is better, stitched maps produce less ghosting, more suitable for the navigation and positioning of robotic systems.

References

[1] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, Robust visual inertial odometry using a direct EKF-based approach, *Proc. IEEE/RSJ International Conf. on Intelligent Robots and Systems,* Hamburg, 2015, 298–304.

[2] T. Qin, P. Li, and S. Shen, Vins-mono: A robust and versatile monocular visual-inertial state estimator, *IEEE Transactions on Robotics, 34*(4), 2018, 1004–1020.

[3] C. Campos, R. Elvira, J.J.G. Rodr'ıguez, and J.M. Montiel, and J.D. Tardós, ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM, *IEEE Transactions on Robotics, 37*(6), 2021, 1874–1890.

[4] J. Ni, X. Wang, and T. Gong, An improved adaptive ORB-SLAM method for monocular vision robot under dynamic environments, *International Journal of Machine Learning and Cybernetics*, 2022, 1–16.

[5] Y. Chen, J. Ni, and E. Mutabazi, A variable radius side window direct SLAM method based on semantic information, *Computational Intelligence and Neuroscience*, 2022, 4075910.

[6] S. Badalkhani, R. Havangi, and M. Farshad, An improved simultaneous localization and mapping for dynamic environments, *International Journal of Robotics and Automation*, *36*(6), 2021, 374–382.

[7] B. Han and L. Xu, MLC-SLAM: Mask loop closing for monocular SLAM, *International Journal of Robotics and Automation, 37*(1), 2022, 107–114.

[8] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Zico Kolter, D. Langer, O. Pink, V. Pratt, M. Sokolsky, G. Stanek, D. Stavens, A. Teichman, M. Werling, and S. Thrun, Towards fully autonomous driving: Systems and algorithms, *Proc. IEEE Intelligent Vehicles Symposium,* Baden-Baden, 2011, 163–168.

[9] J. Saarinen, J. Andreasson, and T. Stoyanov, Normal distributions transform occupancy maps: Application to large-scale online 3D mapping, *Proc. 2013 IEEE International Conf. on Robotics and Automation,* Karlsruhe, 2013, 2233–2238.

[10] Z. Ji and S. Singh, Loam: Lidar odometry and mapping in real-time, *Robotics: Science and Systems Conference, 2*(9), 2014, 1–9.

[11] S. Rusinkiewicz and M. Levoy, Efficient variants of the ICP algorithm, *Proc. of the Third International Conf. on 3-D Digital Imaging and Modeling,* Quebec City, QC, 2001, 145–152.

[12] A. Segal, D. Haehnel, and S. Thrun, Generalized-ICP, *Proceedings of Robotics: Science and Systems, 2*(4), 2009, 435.

[13] M. Magnusson, *The three-dimensional normal-distributions transform: An efficient representation for registration, surface analysis, and loop detection*, Doctoral Dissertation, Örebro universitet, 2009.

[14] W.S. Grant, R.C. Voorhies, and L. Itti, Finding planes in LiDAR point clouds for real-time registration, *Proc. IEEE/RSJ International Conf. on Intelligent Robots and Systems,* Tokyo, 2013, 4347–4354.

[15] A. Geiger, P. Lenz, and R. Urtasun, Are we ready for autonomous driving? The KITTI vision benchmark suite, *Proc. of the IEEE International Conf. on Computer Vision and Pattern Recognition,* Providence, RI, 2012, 3354–3361.

[16] T. Shan and B. Englot, LeGO-LOAM: Lightweight and ground-optimized lidar odometry and mapping on variable terrain, *Proc. 2018 IEEE/RSJ International Conf. on Intelligent Robots and Systems,* Madrid, 2019, 4758–4765.

[17] M. Himmelsbach, F.V. Hundelshausen, and H.J. Wuensche, Fast segmentation of 3D point clouds for ground vehicles, *Proc. of the IEEE Intelligent Vehicles Symposium,* La Jolla, CA, 2010, 560–565.

[18] X. Ji, L. Zuo, C. Zhang, and Y. Liu, LLOAM: LiDAR odometry and mapping with loop-closure detection based correction, *Proc. 2019 IEEE International Conf. on Mechatronics and Automation*, Tianjin, China, 2019, 2475–2480.

[19] J. Lin and F. Zhang, Loam LIVOX: A fast, robust, high-precision lidar odometry and mapping package for LiDARS of small FOV, *Proc. 2020 IEEE International Conf. on Robotics and Automation,* Paris, 2020, 3126–3131.

[20] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, FAST-LIO2: Fast direct LiDAR-inertial odometry, *IEEE Transactions on Robotics*, 2022, 1–21.

[21] S. Hening, C.A. Ippolito, K.S. Krishnakumar, V. Stepanyan, and M. Teodorescu, 3D LiDAR SLAM integration with GPS/INS for UAVs in urban GPS-degraded environments, *Proc. AIAA Information Systems-AIAA Infotech@ Aerospace*, Grapevine, TX, 2017, 448–457.

[22] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, LIO-SAM: Tightly-coupled LiDAR inertial odometry via smoothing and mapping, *Proc. 2020 IEEE/RSJ International Conf. on Intelligent Robots and Systems,* Las Vegas, NV, 2020, 5135–5142.

[23] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, On-manifold preintegration for real-time visual–inertial odometry, *IEEE Transactions on Robotics, 33*(1), 2017, 1–21.

[24] T. Shan, B. Englot, C. Ratti, and D. Rus, LVI-SAM: Tightly-coupled lidar-visual-inertial odometry via smoothing and mapping, *Proc. 2021 IEEE International Conf. on Robotics and Automation,* Xi'an, 2021, 5692–5698.

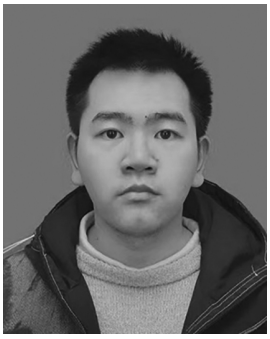
[25] S. Leutenegger, S. Lynen, and M. Bosse, R. Siegwart, and P. Furgale, Keyframe-based visual-inertial odometry using nonlinear optimization, *The International Journal of Robotics Research, 34*(3), 2015, 314–334.

[26] I. Bogoslavskyi and C. Stachniss, Fast range image-based segmentation of sparse 3D laser scans for online operation, *Proc. of the IEEE/RSJ International Conf. on Intelligent Robots and Systems*, Daejeon, 2016, 163–169.

[27] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, ROS: An open-source robot operating system, *IEEE ICRA Workshop on Open Source Software, 3*(3.2), 2009, 5.

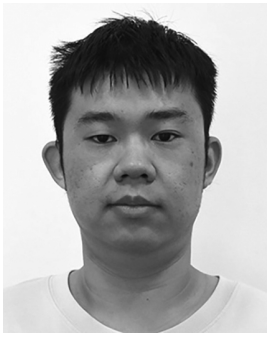
Biographies

Chuanjiang Li received the Ph.D. degree from Shanghai University in 2014. He is currently a Professor with the College of Information, Mechanical and Electrical Engineering, Shanghai Normal University, China. His current research interests include autonomous mobile robots and human–robot interaction.



Ziwei Hu is currently pursuing the master's degree with the Shanghai Normal University. His research focuses on Lidar SLAM and autonomous mobile robots.

Yanfei Zhu received the Ph.D. degree in control science and engineering from East China University of Science and Technology, China, in 2016. He is currently an Associate Professor with the College of Information, Mechanical and Electrical Engineering, Shanghai Normal University, China. His research interests include robot control system, nonlinear systems, and networked control systems.



Xingzhao Ji is currently pursuing the master's degree with Shanghai Normal University. His research interests include robot control system and 3D reconstruction.

Chongming Zhang received the Ph.D. degree in computer applications technology from Fudan University, China, in 2011. He is currently an Associate Professor with the College of Information, Mechanical and Electrical Engineering, Shanghai Normal University, China. His current research interests include human–robot interaction, and embedded system and engineering education technology.

Ziming Qi received the Ph.D. degree from Massey University in 2008. He is currently an Academic Lead with New Zealand Maritime School, Manukau Institute of Technology, New Zealand. His current research interests include the control systems for automotive and marine engineering.