A LiDAR SLAM System With Geometry Feature Group-Based Stable Feature Selection and Three-Stage Loop Closure Optimization

Meng Xu, Shiqi Lin, Jikai Wang, and Zonghai Chen, *Senior Member, IEEE*

***Abstract*— Nowadays, light detection and ranging (LiDAR) sensors have been increasingly used in robotics, particularly in autonomous vehicles, for localization and mapping tasks. However, the use of LiDAR simultaneous localization and mapping (SLAM) in various scenarios is still limited. In this article, we propose a LiDAR SLAM system that addresses this issue by grouping consistent and stable geometry feature to better express the environmental properties in both odometry and loop closure detection. Specifically, to achieve stable geometry feature extraction in the LiDAR odometry component, we adapt an adaptive feature extraction technique that extracts planar, linear, and point features. In addition, we cluster the extracted geometry features to filter out noise. We also analyze the geometry feature matching error and constraint consistency to ensure that the constraints built from these features are stable and repeatable. For the global optimization component, we construct a three-stage loop closure detection approach based on the distribution of geometry feature groups and their corresponding relationships. Quantitative and qualitative experiments on the Karlsruhe Institute of Technology and Toyota Technological Institute (KITTI) dataset, MulRan dataset, and a dataset collected on university campus demonstrate the adaptability, accuracy, and repeatability of our method. In conclusion, our proposed LiDAR SLAM system improves the performance in complex and diverse scenarios by implementing stable geometry feature extraction, effective feature constraint classification, and accurate loop closure detection. The source code of our approach is available at https://github.com/qq1962572025/GeometrySLAM.**

***Index Terms*— Feature group, geometry feature, light detection and ranging (LiDAR) simultaneous localization and mapping (SLAM), loop closure.**

I. Introduction

SIMULTANEOUS localization and mapping (SLAM) is a crucial aspect of several autonomous driving tasks, such as navigation [1], field surveying [2], and mapping [3]. SLAM methods can be categorized into two types, visual SLAM [4], [5] and light detection and ranging (LiDAR) SLAM [6], [7], [8], [9], [10], depending on the sensor used. While visual cameras provide abundant environmental information, they are more sensitive to illumination variations and weakly textured environments. On the other hand, LiDAR sensors offer accurate depth measurements over a wide field of view and can be used in extreme scenarios, leading to higher location accuracy and wider mapping range. Thus, LiDAR SLAM has witnessed significant development in the last decade, and a SLAM system consists of two parts, the front end and the back end, where the former estimates relative poses, and the latter performs global trajectory optimization and map construction.

In order to estimate relative poses, starting from LOAM [6], geometry attributes of local neighbor points are extensively studied to describe the local attributes of features. On the basis of LOAM, lightweight and ground-optimized lidar odometry and mapping (LeGO-LOAM) [7] uses ground detection to extract edge and plane points separately. Implicit moving least squares (IMLS)-SLAM [8] and partition of unity (POU)-SLAM [11] adopt different methods to express plane representation. Planes, lines, and cylinders LiDAR SLAM (PLC-LiSLAM) [12] jointly optimizes PLCs with poses in local and global mapping. Multi-metric linear least square (MULLS) [9] considers six types of geometric features to assemble the linear least square optimization of point-to-point (plane and line) error metrics. Although these state-of-the-art methods perform well on public datasets, their feature extraction strategies are relatively mechanical. The same strategy is usually used for several types of environments, or heuristic parameters need to be manually selected. As a result, most methods are lack of adaptability and generality in real-world widespread application scenarios.

Furthermore, to ensure the global consistency of trajectories, several loop closure detection algorithms have been proposed. Correlation operations on global features based on scan code [13], histogram [14], or segment [15] and pose distances estimated by LiDAR odometry [7] are used to construct global constraints. However, these loop closure detection methods have high requirements on trajectories, resulting in satisfactory results only on close-distance loop closures.

We propose a LiDAR SLAM system to address the aforementioned problems. While application scenarios for autonomous driving exhibit obvious regularity with significant geometry properties, extracted features have a high degree of local consistency and are not randomly distributed. Also, the local regular distribution of point clouds can reflect the type of environment to a certain extent. We extract and cluster planar, linear, and point features into groups, whose related parameters are used to present environmental characteristics. By describing feature group properties, stable features are chosen,

Manuscript received 15 May 2023; revised 14 June 2023; accepted 20 June 2023. Date of publication 6 July 2023; date of current version 17 July 2023. This work was supported by the National Natural Science Foundation of China under Grant 62103393. The Associate Editor coordinating the review process was Dr. Nuria Novas. *(Corresponding author: Zonghai Chen.)*

The authors are with the Department of Automation, University of Science and Technology of China, Hefei 230027, China (e-mail: xm1996@mail.ustc.edu.cn; linshiqi@mail.ustc.edu.cn; wangjk@ustc.edu.cn; chenzh@ustc.edu.cn).

Digital Object Identifier 10.1109/TIM.2023.3292956

and consistence analysis is used to build accurate constraints. Adaptive parameters adjustment in feature extraction and clustering makes the SLAM system adaptability for several types of environments. Based on the proposed consistency weight, multiclass cost function is built and minimized to estimate relative pose using scan-to-scan and scan-to-map registration in geometric ICP. We also propose a three-stage loop closure detection algorithm capable of detecting in a wider range by encoding the relative positional relationship of geometry feature groups to get the similarity between candidate frames, calculating the pose estimation prior by matching geometry feature groups according to their attributes, and employing a pose graph optimization combining relative and loop factors.

In summary, the main contributions of this article are as follows.

1) Starting from the geometry prior knowledge, adaptive selection, clustering, and filtering of features are optimized, and stable feature groups are defined.
2) Multiclass cost function is built and minimized to estimate relative pose using scan-to-scan and scan-to-map registration, and novel proposed consistency weight to combine different geometry features constraints is produced.
3) To handle long-distance loop closures, we propose a new three-stage detection solution based on feature group distribution and alignment, which is more robust to difficult trajectories.

II. Related Works

We review related works from two aspects: front end and back end. For the former, we discuss around the way of feature expression. For the latter, we explore various loop closure detection strategies.

With the development of LiDAR SLAM, geometry feature, semantic feature, and probabilistic feature have been widely used. LOAM [6] extracts edge and surface feature based on curvature and uses the Gauss–Newton method to optimize point-to-line and point-to-plane distances. LeGO-LOAM [7] optimizes feature extraction based on LOAM using techniques, such as projected range image and ground extraction, and introduces loop closure detection and factor graph-based global optimization strategies. IMLS-SLAM [8] is a LiDAR odometry based on sparse sampling and implicit surface representation. Points are sampled by their contribution to the observability of different angles and translations of the vehicle. POU-SLAM [11] uses POU implicit surface to maintain global environment model and merge voxel grids into the model from stage to stage. Guo et al. [16] propose a principal component analysis (PCA)-based [10] curvature evaluation method to extract the edge and planar features, and a normal distribution transform (NDT)-based loop closing is utilized to reduce the cumulative error. MULLS [9] proposed a multiscale linear least squares optimization method based on point, linear, and planar feature, combined with the back end based on neighborhood category context descriptor and truncated least squares estimation and semidefinite relaxation (TEASER) [17], which established a SLAM system suitable for different environments and LiDAR types. PLC-LiSLAM [12] uses PLC adjustment (PLCA) to jointly optimize PLCs poses in front end. In back end, continuous-time registration using PLCs are used to get scan-to-model relative pose.

On the basis of their previous work, SuMa++ [10] introduced RangeNet++ [18] for semantic segmentation. In the process of feature matching error calculation and surfel map update, correction is made according to the consistency of semantic information. Based on generalized iterative closest point (GICP) [19], Voxelized GICP [20] proposes a point cloud registration method based on 3-D voxel distribution to multidistribution. Faster GICP [21] proposed an acceptance–rejection sampling-based two-stage point filter to preserve the points with high planarity that larger matching errors. In [22], the point cloud is described as a set of probability density functions (pdfs) using the NDT algorithm, and then, a rotation-invariant histogram is generated based on the peaks for loop closure detection. The choice of feature type determines the unit of optimization constraint construction and map reconstruction. Composite constraints endow the SLAM algorithm with the ability to migrate and deploy in multiple environments.

LiDAR odometry errors accumulated over time are unavoidable in long trajectories, which can be corrected by loop closure detection. However, loop closure detection is still an open problem for LiDAR SLAM. In recent years, most famous work is the scan context [13], which proposes two-phase matching algorithm. After using the top view of a point cloud to separate ground areas into bins, a rotational-invariant subdescriptor is provided to achieve a feasible search time. Inspired by scan context, intensity scan context [23] is introduced to efficiently integrate both geometry and intensity characteristics into a global signature. LiDAR-Iris [24] calculates similarities as the Hamming distance of two corresponding binary signature images extracted from the two point clouds after several logarithm (LoG)-Gabor filtering and thresholding operations. Continuous-time iterative closest point (CT-ICP) [25], transforms local map into 2-D elevation grid. From this 2-D grid, an elevation image is obtained by clipping the z coordinate of each pixel. Rotation-invariant features are then extracted and compared.

III. System Overview

An overview of the proposed LiDAR SLAM is summarized in Fig. 1. The system inputs are 3-D point clouds, while outputs are six degree of freedom (DOF) pose estimations. The overall system can be divided into front end and back end. During LiDAR odometry, geometry features are extracted, clustered, and filtered from pretreated point cloud. Based on the proposed consistency weight, linear least squares cost function is built and minimized to estimate relative pose using scan-to-scan and scan-to-map registration. At the same time, in the back-end part, under the guide of feature group distribution and alignment, loop closure is detected after certification using scan-to-scan registration. Finally, graph optimization is used to optimize overall trajectory based on relative pose estimation and loop closure detection. The details of these modules are introduced below.

IV. LiDAR Odometry

Our proposed LiDAR odometry is composed of scan-to-scan and scan-to-map registrations. With every input point cloud,

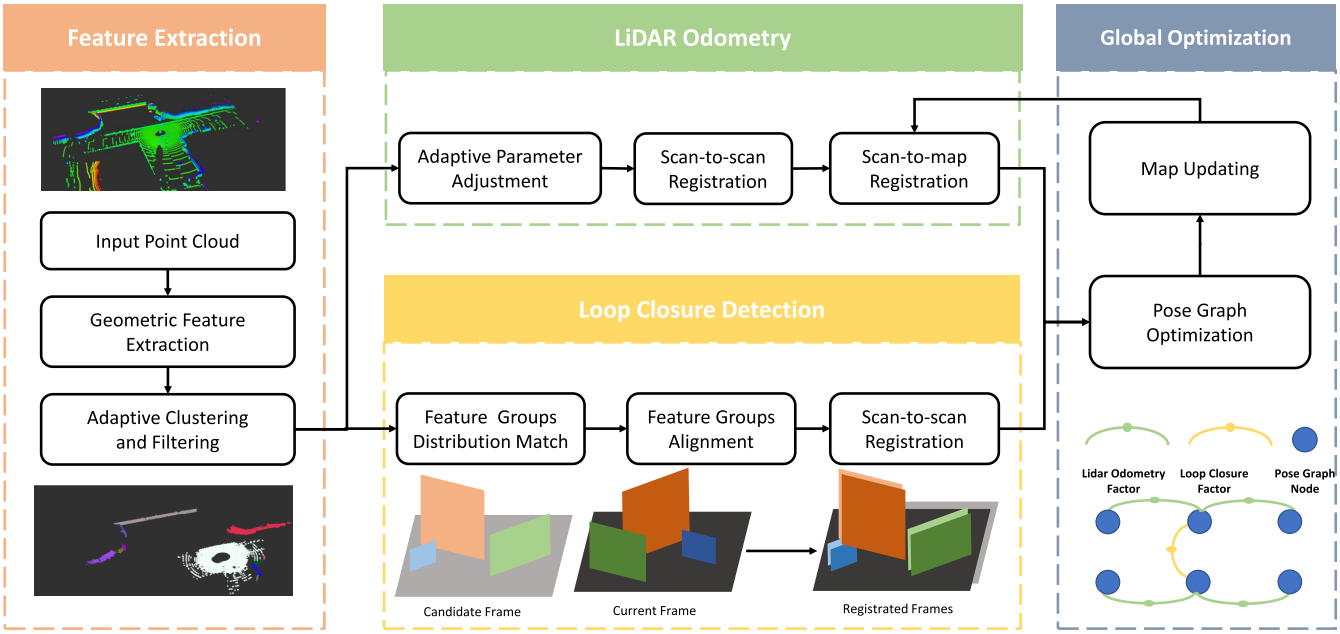


Fig. 1. Overview of our proposed LiDAR system. As can be seen from the left column, stable geometry features are extracted, clustered, and filtered. Adaptive parameter adjustment for cost function is used in both scan-to-scan and scan-to-map registration. Feature group-based three-stage loop closure detection provides loop factors to form a factor graph together with relative factors for global optimization and map updating.

stable geometry features are extracted, clustered, and filtered. Scan-to-scan transformation is then estimated, which providing an initial guess for the following scan-to-map registration. Finally, we update the local map.

A. Geometry Feature Extraction

Point cloud is a sparse representation of environment, whose single measurement only contains coordinates and intensity. However, the local distribution of LiDAR measurements is the embodiment of environment geometric properties and has local consistency. Based on the environmental properties reflected by local neighbor points distribution, we extract three types of geometry feature with adaptive parameters.

1) Downsampling and Neighborhood Estimation: To ensure that point neighborhood contains points from different scans and local point clouds have geometric consistency, we design the following strategy: first, point clouds are divided into ground and nonground points, such as MULLS [9]. According to the distance away from sensor, the ground points are divided into short-distance and large-distance parts at the interval of 30 m. Second, voxel-based downsampling and neighborhood estimation based on nearest neighbor search are used with different parameters. Due to the dense distribution of short-distance ground points, more points are removed with larger proportion than nonground points. Downsampled large-distance ground points query neighborhood points in polar coordinates for higher quality.

2) Geometry Feature Extraction and Clustering: First, we use PCA to analyze the geometry characteristics of feature point. For each downsampled point $\mathbf{p}$, the covariance matrix is calculated based on the neighborhood points estimated above as follows:

$$\mathbf{C} = \frac{1}{N}\Sigma_{i\in N}(\mathbf{p}_i - \bar{\mathbf{p}})(\mathbf{p}_i - \bar{\mathbf{p}})^T. \quad (1)$$

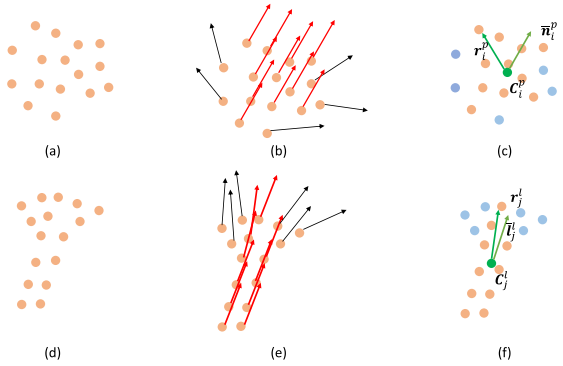


Fig. 2. Overview of building geometry feature group. (a) and (d) Example point clouds. (b) and (e) Normal vector for each point, consistent vectors are colored in red. (c) and (f) Clustered planar or linear feature group and its geometry parameters.

N is the size of neighborhood points $\mathbf{p}_i$, whose average 3-D coordinates are $\bar{\mathbf{p}}$. The eigenvectors $\mathbf{n}_1$, $\mathbf{n}_2$, and $\mathbf{n}_3$ and corresponding eigenvalues $\lambda_1 > \lambda_2 > \lambda_3$ of the covariance matrix $\mathbf{C}$ are then calculated, based on which planarity $\phi_{\text{plane}} = (\lambda_2 - \lambda_3)/\lambda_1$ and linearity $\phi_{\text{line}} = (\lambda_1 - \lambda_2)/\lambda_1$ are defined.

Second, feature points are divided into several types. Among the nonground points, points whose planarity and linearity exceed planarity threshold λ_{pe} or linearity threshold λ_{le} [16] are considered as planar or linear feature points, while only planar feature points are extracted from ground points. Point feature points are sampled from the rest of the points. For planar or linear feature points, normal vectors $\mathbf{n} = [\mathbf{n}_3{}^T \ -\mathbf{n}_3 \cdot \mathbf{p}]^T$ or orientation vectors $\mathbf{l} = \mathbf{n}_1$ are computed. For point feature points, covariance matrix is saved as geometry characteristics.

Finally, feature points that are close in geometry characteristics and location will be expressed as a group. Specifically,

planar point A and planar point B are close when they are neighbors of each other and $\mathbf{n}_A \cdot \mathbf{n}_B > \lambda_{\text{ps}}$. For linear points, geometry characteristic threshold is λ_{ls}. Planar and linear features in the same group share approximate normal and orientation vectors, as shown in Fig. 2. All point features points are saved in one group. For each planar and linear group, we describe them with geometry parameters. For planar feature group i, it has center position $\mathbf{c}_i^p$, mean normal $\bar{\mathbf{n}}_i^p$, mean planarity $\bar{\mathbf{p}}_i^p$, feature size s_i^p, and radius r_i^p, while linear feature group j has mean orientation $\bar{\mathbf{l}}_j^l$ instead.

3) Stable Geometry Feature Selection: The purposes of feature selection are twofold: removing discrete geometry feature with weak expressive ability and downsampling geometry feature with homogenization constraints.

If geometry feature points have discrete geometry characteristics, it will lead to false matches or inconsistent matching cost. So, for linear and planar feature points, small clusters are filtered out, because they are inconsistent in the environmental expression. Feature extracted from similar environment such as from the same large plane always offers similar constraints. Too many repeated constraints are of limited help for solving optimization problems. So, we apply downsampling on repeated geometry feature. For point feature, only random downsampling is required due to their sparse distribution.

The proposed geometry feature selection method can select stable geometry feature with stronger expressive ability and speed up the solution of optimization problems.

4) Adaptive Parameters Adjustment: Geometric feature provides direct expression of the geometric properties of the local environment. In scenes with rich geometric information, high-quality localization can be achieved using only planar features, while in degraded environments, the constraints of point features are essential. Therefore, we roughly estimate the environment properties based on the number and distribution of geometric features and select different feature extraction strategies accordingly. Adaptive parameter tuning speeds up our algorithm and enhances its transferability.

Specifically, when more than 50 planar feature groups are extracted, which represent an environment with a high degree of regularity, these planar feature points are sufficient to provide stable constraints, and the extraction and postprocessing of point and line feature points can be omitted. On the other hand, for environments with fewer than 20 planar and linear feature groups, we resample the entire point cloud to obtain enough point feature points. Furthermore, we modify the parameters of planar and line feature extraction to obtain fewer and more expressive features when the geometric information of the environment is rich. For instance, when more than 100 planar feature groups are extracted, we adjust the planar feature selection threshold λ_{pe} and feature similarity threshold λ_{ps} according to the average planarity and similarity of the extracted planar feature groups. Heuristic threshold parameters used as initial values in experiments are listed as follows:

$$\lambda_{\text{pe}} = 0.70 \tag{2}$$
$$\lambda_{\text{le}} = 0.66 \tag{3}$$
$$\lambda_{\text{ps}} = 0.96 \tag{4}$$
$$\lambda_{\text{ls}} = 0.95. \tag{5}$$

B. Local Map Building and Updating

We maintain a local map consist of extracted feature groups from previous scans. Planar and linear feature groups extracted from each point cloud will try to merge with feature groups in local map, which are spatially and geometrically approximate.

Two planar feature groups a and b will be merged together when two conditions are satisfied

$$\left\|\mathbf{c}_a^p - \mathbf{c}_b^p\right\| < \rho_{\text{plane}}\left(r_a^p + r_b^p\right) \tag{6}$$
$$\bar{\mathbf{n}}_a^p \cdot \bar{\mathbf{n}}_b^p < \lambda_{\text{plane}}. \tag{7}$$

Two linear feature group c and d will be merged together when similar conditions are satisfied

$$\left\|\mathbf{c}_c^l - \mathbf{c}_d^l\right\| < \rho_{\text{line}}\left(r_c^l + r_d^l\right) \tag{8}$$
$$\bar{\mathbf{l}}_c^l \cdot \bar{\mathbf{l}}_c^l < \lambda_{\text{line}} \tag{9}$$

where we set $\rho_{\text{plane}} = 0.9$ and $\rho_{\text{plane}} = 0.8$. Point feature points will just be added together. Filtering and geometry characteristics estimation will be done again for merged feature groups. Every ten frames, local map is cropped with a fixed radius. Distant feature groups will be abandoned.

C. Multiclass Geometric ICP

The inputs of multiclass geometric ICP are target and source feature points, and the outputs are estimated pose and matched feature points. Besides, linear and point feature points participate in ICP optimization only if they are contained in both target and source feature points.

1) Feature Association: Different k-dimensional trees (KDTrees) are constructed for several types of target feature points. In each iteration of ICP, the corresponding points of the same type are queried based on nearest neighbor analysis. For example, source ground planar feature points will query the corresponding point in the KDTree built by target ground planar feature points.

2) Cost Function Composition: We use $\mathbf{p}_{ij}$, $\mathbf{l}_{ij}$, $\mathbf{n}_{ij}$, and $\mathbf{C}_{ij}$ to express position vector, orientation vector, normal vector, and covariance matrix of source or target feature point, while i represents the corresponding feature id and $j = 1$ and 2 represents target or source point cloud. Besides, $\mathbf{n}$, $\mathbf{l}$, and $\mathbf{p}$ represent both homogeneous and nonhomogeneous vector for convenience. $\mathbf{T}$ is the relative pose estimated with rotation matrix $\mathbf{R}$ and translation vector $\mathbf{t}$. Inspired by LeGO-LOAM [7] and MULLS [9], point-to-point, line-to-line, and plane-to-plane error functions in both source to target and target to source alignment are composed and shown in Fig. 3

$$E_{\text{Plane}} = \left(\mathbf{n}_{i2}^T \mathbf{T}^{-1} \mathbf{p}_{i1}\right)^2 + \left(\mathbf{n}_{i1}^T \mathbf{T} \mathbf{p}_{i2}\right)^2 \tag{10}$$
$$E_{\text{Line}} = \|\mathbf{d}_i \times (\mathbf{T}\mathbf{l}_{i2})\|_2 + \|\mathbf{d}_i \times \mathbf{l}_{i1}\|_2 \tag{11}$$
$$E_{\text{Point}} = \|\mathbf{d}_i\|_\sigma \tag{12}$$

where $\mathbf{d}_i = \mathbf{p}_{i1} - \mathbf{T}\mathbf{p}_{i2}$ and $\sigma = (\mathbf{C}_1 + \mathbf{T}\mathbf{C}_2\mathbf{T}^T)^{-1}$. In order to increase the accuracy of error estimation, errors corresponding to planar and linear feature points consist of pairs of point-to-line or point-to-plane distance. Since feature groups express the prior distribution of features in the environment, there are significant correspondences between feature groups. Within feature groups, there is an obvious consistency of correct

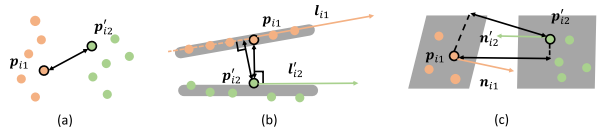


Fig. 3. Schematic of three distinct kinds of distance error functions. (a) Point-to-point. (b) Line-to-line. (c) Plane-to-plane. The superscript $'$ indicates points and vectors after transformation, and black double arrows indicate feature distance vectors.

feature matches, so we set adaptive weights for different feature groups based on consistency. By synthesizing the three feature types, the overall cost function is obtained

$$\mathbf{T} = \underset{\mathbf{T}}{\operatorname{argmin}} \sum_i w^i_{\text{Plane}} \sum_j w^j_{\text{Plane}} E^j_{\text{Plane}} + \sum_i w^{\text{Line}}_i \sum_j w^j_{\text{Line}} E^j_{\text{Line}} + \mu \sum_i E^i_{\text{Point}}. \quad (13)$$

The parameters in (13) are explained as follows: $w_i = \text{num}/\text{size}$, where size is the size of planar or linear feature group and num is the maximum uniform matching subset size. w^j_{Line} and w^j_{Line} express the geometry consistency between features by comparing $\mathbf{l}_{ij}$ and $\mathbf{n}_{ij}$. μ is set to 1 when not enough planar or linear features are extracted; otherwise, it is 0.

3) Solving the Least Squares Problem Based on LM Method: The cost function is represented as a least squares problem using Lie group SE(3). For the minimization, we use Levenberg–Marquardt (LM) and determine increments δ by iteratively solving

$$\delta = \left(\mathbf{J}^T\mathbf{W}\mathbf{J} + \lambda\mathbf{I}\right)\mathbf{J}^T\mathbf{W}\mathbf{r} \quad (14)$$

where $\mathbf{W}$ is a diagonal matrix containing weights mentioned in Section IV-C2 for each term in the cost function, λ is the damping parameter, and $\mathbf{r}$ is the stacked residual vector. To calculate the Jacobian $\mathbf{J}$, we use Lie algebra se(3) and the left Lie derivative to represent and differentiate cost functions. Calculation result of partial derivatives of the left part of plane-to-plane error function is

$$\frac{\partial\left(\mathbf{n}_{i2}^T\mathbf{T}^{-1}\mathbf{p}_{i1}\right)}{\partial\delta\boldsymbol{\xi}} = \begin{bmatrix} -\mathbf{n}_{i2}^T & \mathbf{n}_{i2}^T\left(\mathbf{R}^{-1}\mathbf{p}_{i1} - \mathbf{R}^{-1}\mathbf{t}\right)^\wedge \end{bmatrix} \quad (15)$$

where $\delta\boldsymbol{\xi} = [\delta\boldsymbol{\rho}, \delta\boldsymbol{\phi}]$ is Lie algebra representation of left perturbation. Similarly, we can get the partial derivatives of other error functions

$$\frac{\partial\left(\mathbf{n}_{i1}^T\mathbf{T}\mathbf{p}_{i2}\right)}{\partial\delta\boldsymbol{\xi}} = \begin{bmatrix} \mathbf{n}_{i1}^T & -\mathbf{n}_{i1}^T(\mathbf{R}\mathbf{p}_{i2} + \mathbf{t})^\wedge \end{bmatrix} \quad (16)$$

$$\frac{\partial\left(-(\mathbf{T}\mathbf{l}_{i2})^\wedge\mathbf{d_i}\right)}{\partial\delta\boldsymbol{\xi}} = \mathbf{l}_{i2}^\wedge\begin{bmatrix} \mathbf{I} & -\left(\mathbf{R}^{-1}(\mathbf{p_{i1}} - \mathbf{t})\right)^\wedge \end{bmatrix} \quad (17)$$

$$\frac{\partial\left(-\mathbf{l}_{i1}^\wedge\mathbf{d_i}\right)}{\partial\delta\boldsymbol{\xi}} = \mathbf{l}_{i1}^\wedge\begin{bmatrix} \mathbf{I} & -(\mathbf{R}\mathbf{p}_{i2} + \mathbf{t})^\wedge \end{bmatrix} \quad (18)$$

$$\frac{\partial(\mathbf{d_i})}{\partial\delta\boldsymbol{\xi}} = -\begin{bmatrix} \mathbf{I} & -(\mathbf{R}\mathbf{p}_{i2} + \mathbf{t})^\wedge \end{bmatrix}. \quad (19)$$

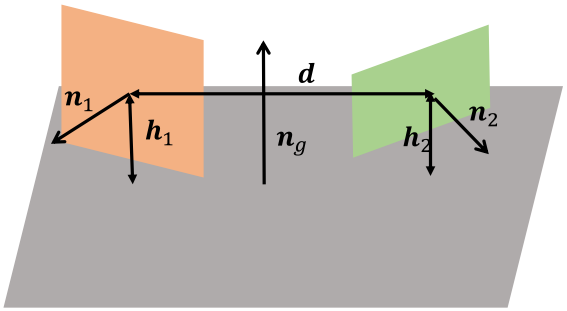


Fig. 4. Schematic of relative positional relationship between feature groups, use planar feature group as an example. $\mathbf{n}_1$, $\mathbf{n}_2$, and $\mathbf{n}_g$ are normal vectors; $\mathbf{h}_1$ and $\mathbf{n}_2$ are vertical line to the ground from center points; and $\mathbf{d}$ is vector between center points.

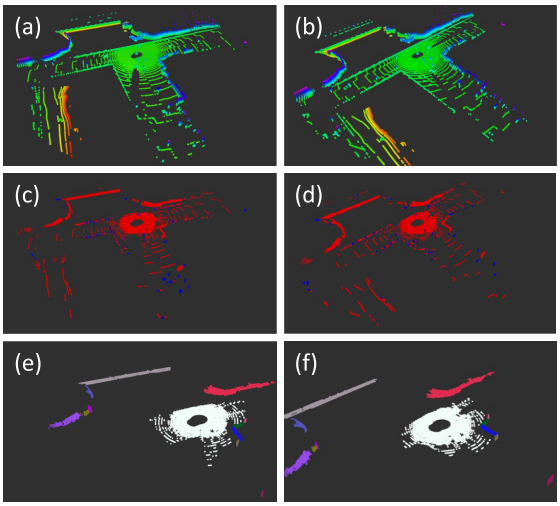


Fig. 5. Overview of feature group building and matching in current and candidate point clouds. (a) and (b) Point clouds. (c) and (d) Extracted planar (red) and linear (blue) feature points. (e) and (f) Matched feature groups. Colors indicating corresponding relationships.

V. Loop Closure Detection and Graph Optimization

Although our proposed algorithm can accurately estimate the relative pose of mobile platform, LiDAR odometry still suffers from drift error in long-duration navigation tasks. Loop closure algorithm is designed to solve the problem. As shown in Fig. 1, our three-stage loop closure detection program run every five point clouds after scan-to-map registration. First, we use distribution of feature groups to search for loop closure candidate. Then, comparing geometry parameters, feature groups find their corresponding matches and get initial transform. Finally, scan-to-scan registration is done. Verified loop closure constraints will be added to the graph optimization to optimize overall trajectory.

A. Find Loop Closure Candidate

Since local LiDAR measurement and extracted geometry features are lack of discrimination and different feature groups describing the same area inevitably have errors in parameters, such as radius and center position, it is too difficult to

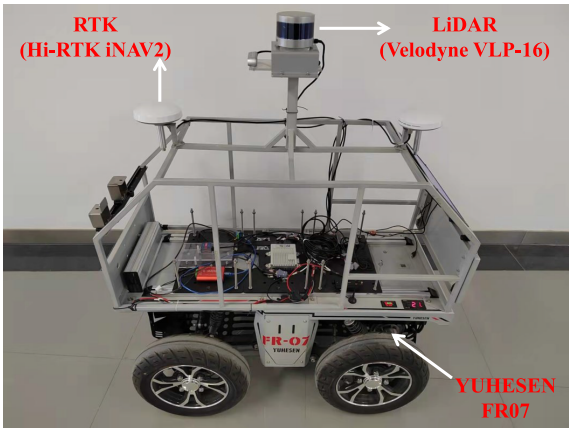


Fig. 6. Our data acquisition platform YUHESEN FR07.

directly establish constrains between feature groups. However, the relative position of multiple feature groups has sufficient expressiveness. Due to the fact that ground feature group has strong discrimination and can be stably extracted, we use 11-D vectors to describe the relative positional relationship between two nonground feature groups and ground feature group, as shown in Fig. 4

$$\mathbf{b} = [N_G \ N_{N1} \ N_{N2} \ H_1 \ H_2 \ D_P \ L_1 \ L_2 \ L_3 \ \mathrm{ID}_1 \ \mathrm{ID}_2]. \quad (20)$$

Among them, N_G, N_{N1}, and N_{N2} are the number of features contained in feature groups; $H_1 = \|\mathbf{h}_1\|$, $H_2 = \|\mathbf{h}_2\|$, and $D_P = \|\mathbf{d}\|$ are vertical distance to ground of nonground feature groups and distance between center positions; and L_1, L_2, and L_3 express relative relationships based on the normal or orientation vectors using dot multiplication. ID_1 and ID_2 record the sequence numbers of nonground features groups for subsequent feature group alignment. We use the weighted Euclidean distance sum to define the matching cost $d_{ij,kl}$, where ij and kl are sequence numbers for descriptors, and the weights of numbers, distances, and relative relationships descend by orders of magnitude in turn. ID_1 and ID_2 are only for marking. For each point cloud, nonground feature groups whose size is above threshold combine in pairs and generate descriptor with ground feature group, which is stored in map dataset.

When calculating matching cost, for each feature group descriptor in current point cloud, we search for their smallest matching cost in query point cloud. Total cost between current and query point cloud is set as the average value of all descriptor matching costs. When searching for loop closure candidate, point cloud with the smallest matching cost queried in the map data is set as a loop closure candidate.

B. Verify Loop Closure Constraint

We use scan-to-scan registration to verify loop closure constraint. In order to speed up and expand the convergence range when optimizing relative transform, the initial estimation is calculated after matching feature groups. Fig. 5 demonstrates the process.

For current point cloud and candidate point cloud, excluding ground and point feature groups, we construct a matrix $N \in R^{n\times m}$ to describe the correlation between every feature group, where n and m are the number of nonground feature groups of two point clouds. Each element $N_{i,j} = \Sigma_{k=1}^{n}\Sigma_{l=1}^{m} d_{ik,jl}$ is determined by the corresponding feature group pairs matching cost. By doing this, we transform feature group matching into maximum cardinality matching in graph theory. Then, we use the Hungarian algorithm [28] to obtain relationships between feature groups.

Based on matching relationships and feature group geometric description, we use the scan-to-scan registration with fixed corresponding to obtain the initial position. Considering all geometric features in current point cloud and candidate point cloud, relative position is calculated based on scan-to-scan registration with initial value to verify loop closure.

C. Graph Optimization

Taking loop closure constraints as loop factors and adjacent transformation estimated by the LiDAR odometry as relative factors, we build a factor graph model, and an incremental smoothing and mapping (iSAM) [29] method is used to solve the graph optimization problem.

VI. Experiments

We design a series of quantitative and qualitative experiments to demonstrate the effectiveness of our algorithm, which are performed on three datasets with more than 20 trajectories. All the experiments are conducted on a PC equipped with an Intel i5-10600KF CPU and 16-GB RAM.

A. Datasets

The Karlsruhe Institute of Technology and Toyota Technological Institute (KITTI) dataset [30] is composed of 11 sequences with ground-truth trajectories provided by global navigation satellite system and inertial navigation system (GNSS-INS). A Velodyne HDL-64E LiDAR is used to collect data in multiple environments, including the urban, country, and highway scene.

The MulRan dataset [31] includes radar and LiDAR data with great structural and temporal diversity specifically targeting the urban environment. Data are collected by Ouster OSI-64 and Navtech CIR204-H, whose 6-D ground-truth trajectory is made via SLAM using fiber optic gyro (FOG), virtual reference station GPS (VRS-GPS), and ICP.

Acquired in the campus of University of Science and Technology of China (USTC) using our own vehicle with a Velodyne VLP-16, our collected dataset is composed of nine sequences with 65 789 scans and sequences totaling 10 610-m long, which contain structural, unstructured scenes and loop closures with different distances. The data acquisition platform we use is YUHESEN FR07, which is shown in Fig. 6. With Hi-target iNAV2 real-time kinematic (RTK) equipped, we use the calibrated GPS integrated navigation system (GINS) data as the ground-truth poses. The extrinsic calibration between LiDAR and GINS is estimated by lidar_align.[1]

B. SLAM Experiments

Quantitative experiment is done on both datasets. Following the result evaluation criterion [30], the average translational

[1] https://github.com/ethz-asl/lidar_align

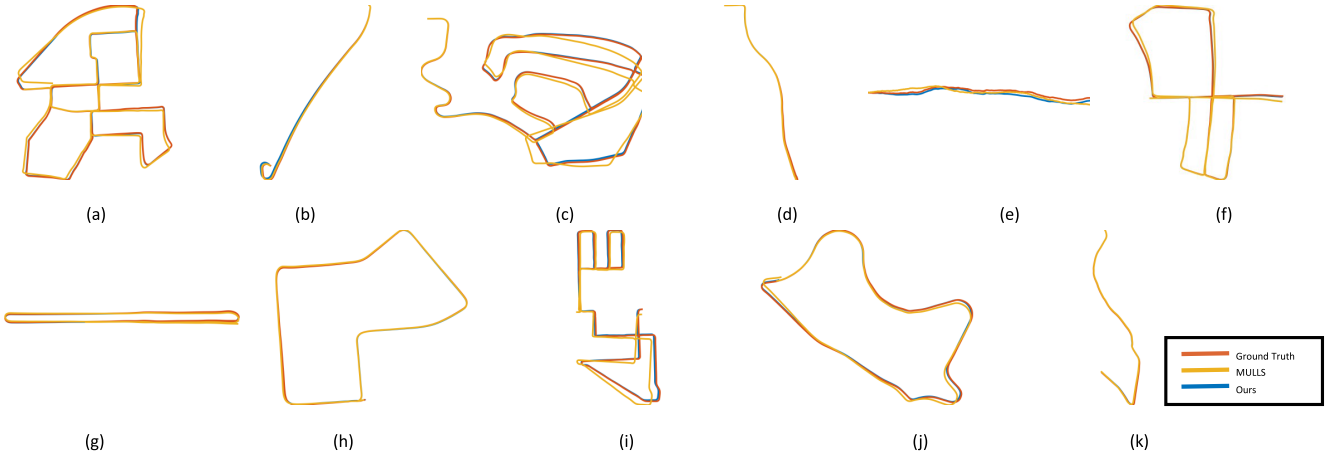


Fig. 7. Estimated trajectories compare with MULLS on the KITTI dataset. Ground-truth trajectories are also presented. (a) Seq 00. (b) Seq 01. (c) Seq 02. (d) Seq 03. (e) Seq 04. (f) Seq 05. (g) Seq 06. (h) Seq 07. (i) Seq 08. (j) Seq 09. (k) Seq 10.

TABLE I

ATE [%] EVALUATION AND COMPARISON ON THE KITTI DATASET

	00	01	02	03	04	05	06	07	08	09	10	Mean
SuMa++ [10]	0.64	1.60	1.00	0.67	0.37	0.40	0.46	0.34	1.10	0.47	0.66	0.70
ROI-cloud [26]	1.23	14.0	7.01	1.18	0.36	0.76	0.86	1.09	3.68	1.28	1.47	2.99
DGP-SLAM [27]	0.85	1.94	0.99	0.79	0.71	0.42	0.44	0.44	1.00	0.73	1.09	0.85
PLC-LiSLAM [12]	0.61	1.38	0.82	0.52	0.36	0.38	0.42	0.41	0.73	0.61	0.67	0.63
MULLS [9]	0.54	**0.62**	0.69	0.61	0.35	0.29	0.29	0.27	0.83	0.51	0.61	0.51
Ours	**0.26**	0.98	**0.31**	**0.45**	**0.30**	**0.17**	**0.21**	**0.21**	**0.54**	**0.40**	**0.30**	**0.38**

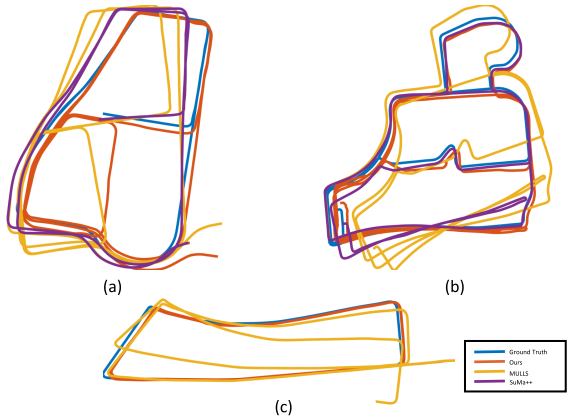


Fig. 8. Estimated trajectories compare with MULLS and SuMa++ on the MulRan dataset. Ground truth trajectories are also presented. (a) DCC 01. (b) KAIST 01. (c) Riverside 01.

TABLE II

ATE [%] EVALUATION AND COMPARISON ON THE MULRAN DATASET. - INDICATES TRACKING FAILURE

Methods	DCC01	KAIST01	Riverside01	Mean
MULLS [9]	3.44	4.09	6.61	4.71
SuMa++ [10]	5.16	5.39	-	5.28
ours	**1.38**	**1.58**	**2.00**	**1.66**

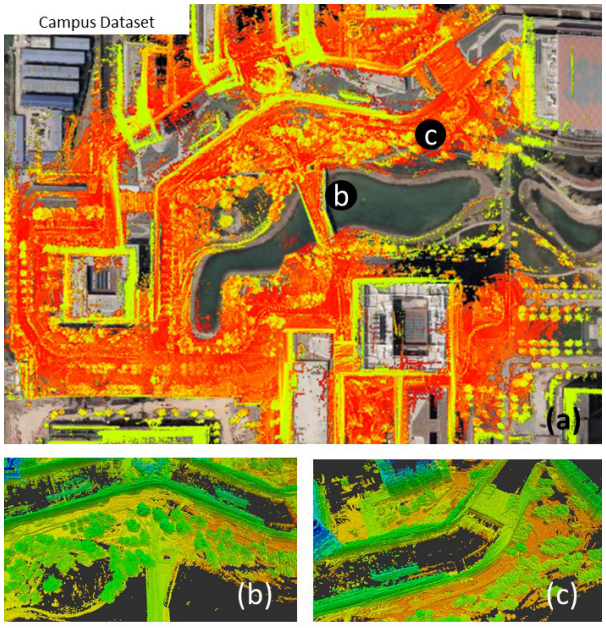


Fig. 9. Overview of our results on Campus 220714 sequence. (a) Cloud map aligned with Google Earth. (b) and (c) Example of the generated point cloud map.

error (ATE) is used for localization accuracy evaluation. To compare our method, we evaluated five start-of-the-art SLAM systems on the KITTI dataset: SuMa++ [10], region of interest cloud (ROI-cloud) [26], directed geometry point (DGP)-SLAM [27], PLC-LiSLAM [12], and MULLS [9]. Since some of these methods do not provide source code, results of the KITTI dataset are taken from their original papers. On the MulRan dataset, SuMa++[2] and MULLS[3] are compared. Since the semantic model cannot be migrated to point clouds collected by Velodyne VLP-16, only MULLS

[2]https://github.com/PRBonn/semantic_suma
[3]https://github.com/YuePanEdward/MULLS

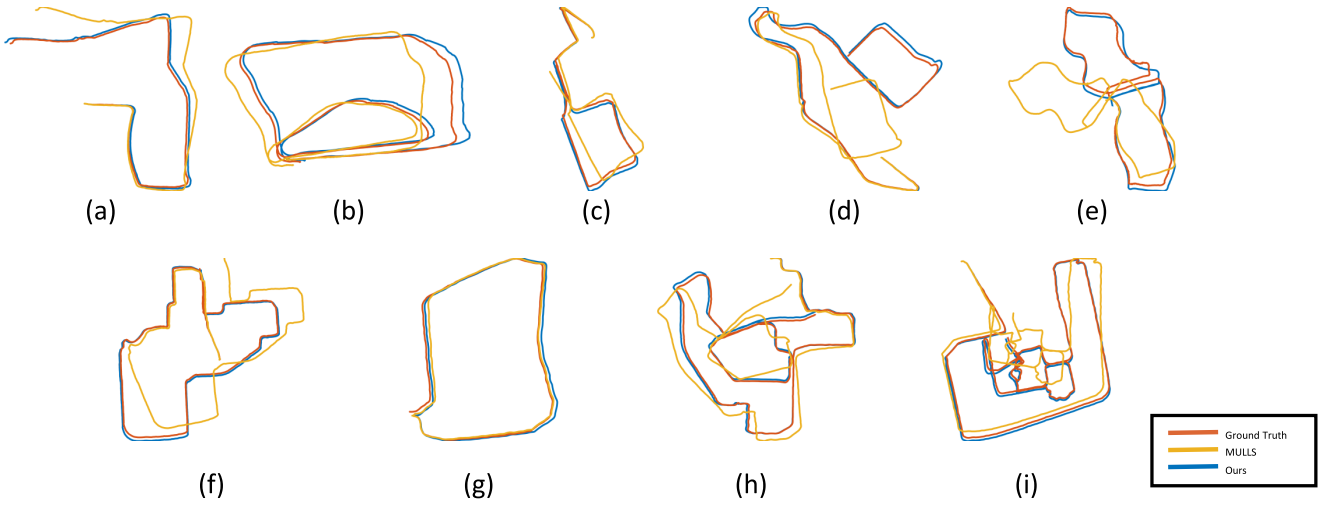


Fig. 10. Estimated trajectories compare with MULLS on the Campus dataset. Ground-truth trajectories are also presented. (a) 211101. (b) 211102. (c) 2111103. (d) 220101. (e) 220102. (f) 220103. (g) 220628. (h) 220714. (i) 220718.

TABLE III

ATE [%] EVALUATION AND COMPARISON ON THE CAMPUS DATASET

Sequence	MULLS	Ours	Sequence	MULLS	Ours
211101	11.5	**1.79**	220103	9.94	**1.40**
211102	3.10	**0.93**	220628	2.75	**1.23**
211103	6.66	**1.93**	220714	5.39	**1.01**
220101	12.2	**3.43**	220718	4.71	**0.90**
220102	8.96	**2.23**	Mean	7.25	**1.65**

TABLE IV

MEDIAN AND STANDARD DEVIATION OF ATE [%]

Sequence	KITTI 00	DCC 01	Campus 220714
Median	0.26	1.38	1.01
Standard Deviation	0.00005	0.0002	0.0005

is evaluated on collected Campus dataset. MULLS is run at corresponding preset parameters on different sequences.

On the KITTI dataset, the performance of all methods is reported in Table I. The estimated trajectories are also given in Fig. 7. The environmental characteristics of different trajectories in KITTI have obvious differences from villages to highways. MULLS designs three different sets of parameters, while we use only one. It is obvious that our method achieves small ATE on most sequence, which demonstrating the accuracy of our LiDAR SLAM system and the effectiveness of our feature extraction and parameter adaptive adjustment.

On the MulRan dataset, trajectories and their evaluation results are shown in Table II and Fig. 8. Due to the occlusion of LiDAR sensor, each single point cloud contains only two-thirds of the theoretical value. Coupled with frequent reverse loop closures, the difficulty of these trajectories is significantly improved, while our method achieves the best results on all trajectories.

Since the usage of Velodyne VLP-16, we collected more sparse data than datasets above. Furthermore, we design trajectories to pass through buildings, cross the bridge, and then extend to lakeside. As a result, our collected dataset is more challenging due to the sensor type and trajectory design. Experiments are shown in Table III and Fig. 9. The mapping

TABLE V

ABLATION STUDY ATE [%] WITH RESPECT TO ADAPTIVE FEATURE EXTRACTION AND COST FUNCTION PARAMETERS USAGE

Sequence	KITTI 00	DCC 01	Campus 220714
Both Adaptive Parameters Usage	**0.26**	**1.38**	**1.01**
Only Adaptive Feature Extraction Parameters Usage	0.44	1.90	2.59
Only Adaptive Cost Function Parameters Usage	0.29	1.52	1.33
Both Fixed Parameters Usage	0.52	2.06	2.74

results of one area in our campus are shown in Fig. 10. We can tell that our mapping results demonstrate less distortions compared with that of the MULLS. Performance on these different datasets proves the transferability of our method.

In order to verify the repeatability, we run five times for each sequence and show the median results in our experiments. The standard deviations and medians of some sequences are shown in Table IV. It is evident that the standard deviation is relatively larger on the Campus dataset. This can be attributed to the dataset's inherent difficulty, particularly in degraded scenarios where point feature is the sole reliance in multiclass geometric ICP. In these cases, downsampling introduces a relatively higher level of randomness due to the absence of feature group constraints. From another perspective, the ability to adapt to several types of environments demonstrates the adaptability of our proposed method.

C. Adaptive Parameters Studies

In this section, we perform ablation studies to validate the performance of adaptive parameters adjustment. Experiments are done in all three datasets, as shown in Table V. Adaptive geometry extraction parameters are used to select stable and complete features, which leads to accurate trajectories and less features, and adaptive cost function is proven effective and is more obvious in improving performance due to the ability to suppress inconsistent feature matching. In all sequences, the usage of adaptive parameters is proven effective. Specifically, ATE realizes obvious improvement on the Campus dataset

TABLE VI

PROPORTION OF DIFFERENT LOOP CLOSURE DISTANCE INTERVALS (UNIT: m) IN KITTI AND CAMPUS DATASETS

Loop closure interval	0-1	1-2	2-3	3-4	4-5
KITTI dataset	0.50	0.22	0.12	0.11	0.05
Campus dataset	0.28	0.12	0.11	0.12	0.37

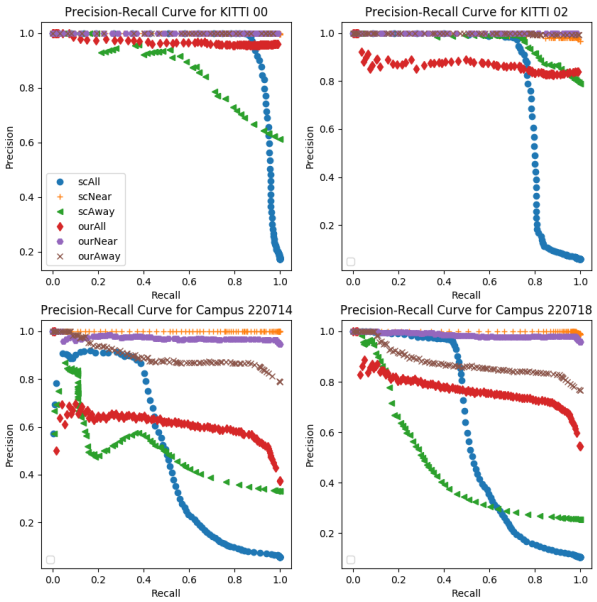


Fig. 11. Precision–recall curves for the evaluation trajectories.

TABLE VII

PER FRAME EFFICIENCY (UNIT: ms) AND POINT NUMBER ANALYSIS IN DETAIL

Module	KITTI	MulRan	Campus
Feature Extraction	16.54	14.62	2.64
Scan-to-scan Registration	3.86	2.57	1.46
Scan-to-map Registration	5.02	3.99	1.93
Graph optimization	2.24	1.84	1.18
Points number	2182.2	1527.8	1076.1

with the consideration of point features. At the same time, the significant difference in the distribution of feature types also reflects the corresponding environmental properties. In MulRan DCC 01 sequence, farther away buildings combined with more trees leading to more extracted linearity features and less planarity features than the KITTI 00 sequence. In Campus 220714 sequence, because the collection environment includes courtyards, there are areas where geometric information is insufficient and needs to be compensated by point features. This also proves the correctness and necessity of designing adaptive parameters by inferring environmental properties based on features.

D. Loop Closure Experiments

Similar to the definition in scan context [13], we regard the case where distances between trajectories are less than 5 m as loop closures. Furthermore, we refer to loop closures within 2 m as close distance, while the rest as long distance. Since most of the loop closures in the KITTI dataset are generated by passing through the same two-lane road, the close-distance loop closure distances account for the majority. In many real applications, robots work in large-scale scenarios that require long-distance loop detection. Therefore, we deliberately designed this type of scene when collecting trajectories. We show the detailed distribution in Table VI. Each value in the table represents the proportion of the corresponding interval to the total loop closures. Most KITTI loop closure distances are less than 2 m, while the loop closure detection of our data is biased toward long-distance scenarios.

We compared with scan context on both datasets, and the precision–recall curve is shown in Fig. 11. To effectively demonstrate the loop closure performance at different distances, we draw three types of precision–recall curves for each sequence in order to show general performance in close-distance loops, long-distance loops, and all frames. For example, legend scAway show the precision–recall curve of scan-context method on long-distance loops. Scan context does great job in close distance and performs poorly in long distance, while our method maintains satisfactory performance, thus achieving better results overall.

E. Efficiency Analysis

A detailed analysis of each part of our algorithm is shown in Table VII. In the KITTI dataset, each frame takes 30 ms on average with feature points average size of nearly 2000. We extract less feature points and run faster in the MulRan dataset because of their occlusion point clouds. For 16-line point cloud, each frame takes 10 ms on average with nearly 1000 average feature points. Our method operates over 30 Hz on all trajectories and can run in real time on a moderate PC.

VII. CONCLUSION

In this article, we propose a LiDAR SLAM based on stable geometry feature extraction and three-stage loop closure detection under environmental properties expression using feature groups. By grouping consistent and stable geometry feature in both odometry and loop closure detection, our method can better express the environmental properties and have adaptive, accurate, and repeatable real-time performance proved by the experiment results on various datasets with challenging scenarios. In future work, we plan to analyze the contribution of different feature extraction strategies to the pose estimation accuracy and further improve the adaptability of our algorithm in complex environments.

REFERENCES

[1] H. Temeltas and D. Kayak, "SLAM for robot navigation," *IEEE Aerosp. Electron. Syst. Mag.*, vol. 23, no. 12, pp. 16–19, Dec. 2008.

[2] K. Ebadi et al., "LAMP: Large-scale autonomous mapping and positioning for exploration of perceptually-degraded subterranean environments," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 80–86.

[3] R. A. Newcombe et al., "KinectFusion: Real-time dense surface mapping and tracking," in *Proc. 10th IEEE Int. Symp. Mixed Augmented Reality*, Oct. 2011, pp. 127–136.

[4] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: A versatile and accurate monocular SLAM system," *IEEE Trans. Robot.*, vol. 31, no. 5, pp. 1147–1163, Oct. 2015.

[5] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 3, pp. 611–625, Mar. 2018.

[6] J. Zhang and S. Singh, "LOAM: LiDAR odometry and mapping in real-time," *Robot., Sci. Syst.*, vol. 2, no. 9, pp. 1–9, 2014.

[7] T. Shan and B. Englot, "LeGO-LOAM: Lightweight and ground-optimized LiDAR odometry and mapping on variable terrain," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 4758–4765.

[8] J.-E. Deschaud, "IMLS-SLAM: Scan-to-model matching based on 3D data," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 2480–2485.

[9] Y. Pan, P. Xiao, Y. He, Z. Shao, and Z. Li, "MULLS: Versatile LiDAR SLAM via multi-metric linear least square," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 11633–11640.

[10] X. Chen, A. Milioto, E. Palazzolo, P. Giguère, J. Behley, and C. Stachniss, "SuMa++: Efficient LiDAR-based semantic SLAM," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 4530–4537.

[11] J. Jiang, J. Wang, P. Wang, and Z. Chen, "POU-SLAM: Scan-to-model matching based on 3D voxels," *Appl. Sci.*, vol. 9, no. 19, p. 4147, Oct. 2019. [Online]. Available: https://www.mdpi.com/2076-3417/9/19/4147

[12] L. Zhou, G. Huang, Y. Mao, J. Yu, S. Wang, and M. Kaess, "PLC-LiSLAM: LiDAR SLAM with planes, lines, and cylinders," *IEEE Robot. Autom. Lett.*, vol. 7, no. 3, pp. 7163–7170, Jul. 2022.

[13] G. Kim and A. Kim, "Scan context: Egocentric spatial descriptor for place recognition within 3D point cloud map," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 4802–4809.

[14] T. Röhling, J. Mack, and D. Schulz, "A fast histogram-based similarity measure for detecting loop closures in 3-D LiDAR data," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2015, pp. 736–741.

[15] R. Dubé, D. Dugas, E. Stumm, J. Nieto, R. Siegwart, and C. Cadena, "SegMatch: Segment based place recognition in 3D point clouds," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 5266–5272.

[16] S. Guo, Z. Rong, S. Wang, and Y. Wu, "A LiDAR SLAM with PCA-based feature extraction and two-stage matching," *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1–11, 2022.

[17] H. Yang, J. Shi, and L. Carlone, "TEASER: Fast and certifiable point cloud registration," *IEEE Trans. Robot.*, vol. 37, no. 2, pp. 314–333, Apr. 2021.

[18] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, "RangeNet++: Fast and accurate LiDAR semantic segmentation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 4213–4220.

[19] A. V. Segal, D. Haehnel, and S. Thrun, "Generalized-ICP," *Robot. Sci., Syst.*, vol. 2, no. 4, p. 435, Jun. 2009.

[20] K. Koide, M. Yokozuka, S. Oishi, and A. Banno, "Voxelized GICP for fast and accurate 3D point cloud registration," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 11054–11059.

[21] J. Wang, M. Xu, F. Foroughi, D. Dai, and Z. Chen, "FasterGICP: Acceptance-rejection sampling based 3D LiDAR odometry," *IEEE Robot. Autom. Lett.*, vol. 7, no. 1, pp. 255–262, Jan. 2022.

[22] Q. Meng, H. Guo, X. Zhao, D. Cao, and H. Chen, "Loop-closure detection with a multiresolution point cloud histogram mode in LiDAR odometry and mapping for intelligent vehicles," *IEEE/ASME Trans. Mechatronics*, vol. 26, no. 3, pp. 1307–1317, Jun. 2021.

[23] H. Wang, C. Wang, and L. Xie, "Intensity scan context: Coding intensity and geometry relations for loop closure detection," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 2095–2101.

[24] Y. Wang, Z. Sun, C. Xu, S. E. Sarma, J. Yang, and H. Kong, "LiDAR iris for loop-closure detection," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 5769–5775.

[25] P. Dellenbach, J. Deschaud, B. Jacquet, and F. Goulette, "CT-ICP: Real-time elastic LiDAR odometry with loop closure," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2022, pp. 5580–5586.

[26] Z. Zhou, M. Yang, C. Wang, and B. Wang, "ROI-cloud: A key region extraction method for LiDAR odometry and localization," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 3312–3318.

[27] S. Liang, Z. Cao, C. Wang, and J. Yu, "A novel 3D LiDAR SLAM based on directed geometry point and sparse frame," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 374–381, Apr. 2021.

[28] H. W. Kuhn, "The Hungarian method for the assignment problem," *Nav. Res. Logistics Quart.*, vol. 2, nos. 1–2, pp. 83–97, Mar. 1955.

[29] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "iSAM2: Incremental smoothing and mapping using the Bayes tree," *Int. J. Robot. Res.*, vol. 31, no. 2, pp. 216–235, Feb. 2012.

[30] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 3354–3361.

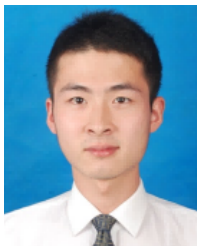
[31] G. Kim, Y. S. Park, Y. Cho, J. Jeong, and A. Kim, "MulRan: Multimodal range dataset for urban place recognition," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 6246–6253.

Meng Xu received the B.S. degree from the University of Science and Technology of China (USTC), Hefei, China, in 2018, where he is currently pursuing the Ph.D. degree with the Department of Automation.

His research interests include robotics and simultaneous localization and mapping (SLAM).

Shiqi Lin received the B.S. degree from Dalian Minzu University, Dalian, China, in 2017. He is currently pursuing the Ph.D. degree with the Department of Automation, University of Science and Technology of China (USTC), Hefei, China.

His research interests include state estimation, visual localization, and semantic scene understanding.



Jikai Wang received the B.S. degree from the University of Yanshan, Qinhuangdao, China, in 2014.

He is currently a Post-Doctoral Researcher with the Department of Automation, University of Science and Technology of China (USTC), Hefei, China. His research interests include knowledge representation, intelligent information processing, robotics, visual simultaneous localization and mapping (SLAM), and machine learning.

Zonghai Chen (Senior Member, IEEE) was born in Anhui, China, in 1963. He received the B.S. degree in automation and the M.E. degree in control theory and control engineering from the University of Science and Technology of China (USTC), Hefei, China, in 1988 and 1991, respectively.

He has been a Professor with the Department of Automation, USTC, since 1998. His research interests include modeling and control of complex systems, intelligent robotic and information processing, energy management technologies for electric vehicles, and smart microgrids.

Prof. Chen is a member of the Robotics Technical Committee and the Modeling, Identification and Signal Processing Technical Committee of the International Federation of Automation Control (IFAC). He was a recipient of special allowances from the State Council of China.