T-LOAM: Truncated Least Squares LiDAR-Only Odometry and Mapping in Real Time

Pengwei Zhou, Xuexun Guo, Xiaofei Pei, and Ci Chen

***Abstract*— We propose a novel, computationally efficient, and robust light detection and ranging (LiDAR)-only odometry framework based on truncated least squares termed T-LOAM. Our method focuses on alleviating the impact of outliers to allow robust navigation in sparse, noisy, or cluttered scenarios where degeneration occurs. As preprocessing, the multiregion ground extraction and dynamic curved-voxel clustering methods are proposed to accomplish the segmentation of 3D point clouds and filter out unstable objects. A novel feature extraction module is tailored to discriminate four peculiar features: edge features, sphere features, planar features, and ground features. As frontend, a hierarchical feature-based LiDAR-only odometry performs precise motion estimates through the truncated least squares method for directly processing various features. The preprocessing model and motion estimation precision have been evaluated on the KITTI odometry benchmark as well as various campus scenarios. The experimental results have demonstrated the real-time capability and superior precision of the proposed T-LOAM over other state-of-the-art algorithms.**

***Index Terms*— Dynamic curved-voxel clustering (DCVC), multiregion ground extraction (MRGE), T-LOAM, truncated least squares (TLS).**

NOMENCLATURE

t_k	Scan-end time of the kth LiDAR scan.
L_k	LiDAR body frame at the time t_k.
Q_i, S_j	ith quadrant and jth section for MRGE.
$\mathbb{F}_k^v$, $\mathbb{F}_k^b$	Foreground and background from MRGE at the time t_k.
$\mathbb{F}_k^d$	DCVC model output from $\mathbb{F}_k^v$ at the time t_k.
$\mathbb{F}_k$	All the features from the current scan at the time t_k in L_k.
F_k^e, F_k^s, F_k^p, F_k^g	Edge, sphere, planar, and ground features of $\mathbb{F}_k$ in L_k, respectively.
$\mathbb{M}_k^w$	kth features of submap at the time t_k in world frame.
x_k^w	Robot state in the global frame at the time t_k.

Manuscript received March 21, 2021; revised April 28, 2021; accepted May 20, 2021. Date of publication June 3, 2021; date of current version January 12, 2022. This work was supported in part by NSFC under Grant 51505354. *(Corresponding author: Xiaofei Pei.)*

Pengwei Zhou is with the Department of Automotive Engineering, Wuhan University of Technology, Wuhan 430070, China (e-mail: zpw6106@gmail.com).

Xuexun Guo and Ci Chen are with the School of Automotive Engineering, Wuhan University of Technology, Wuhan 430070, China, and also with the Hubei Key Laboratory of Advanced Technology of Automotive Components, Wuhan University of Technology, Wuhan 430070, China (e-mail: guo6531@163.com; chenc1520@whut.edu.cn).

Xiaofei Pei is with the Hubei Research Center for New Energy and Intelligent Connected Vehicle, Wuhan University of Technology, Wuhan 430070, China (e-mail: peixiaofei7@whut.edu.cn).

Data is available on-line at https://github.com/zpw6106/tloam.git.

Digital Object Identifier 10.1109/TGRS.2021.3083606

I. INTRODUCTION

THE tremendous development of mobile robot and autonomous vehicle technologies jointly promote the progress of simultaneous localization and mapping (SLAM) algorithm, which can provide accurate pose and environment information.

Compared with visual sensors, 3D light detection and ranging (LiDAR) can provide lighting-invariant and precise perception of the environments with long detection range and superior robustness. Thus, they are broadly employed as efficient and stable sensors in autonomous driving scenarios. However, 3D point clouds are generic of disequilibrium and inhomogeneity containing a large of outliers, which will significantly deteriorate the precision of LiDAR-only odometry. Although additional navigation sensors such as IMU and GNSS can be tailored to correct odometry drift, they still cannot provide precise enough position information in some GNSS signal reception constrained scenarios (e.g., tunnels, tall buildings, and parking garages). To solve this problem, we focus on 3D point clouds segmentation and extract stable environmental characteristics, thus improving the precision of LiDAR-only odometry in the presence of a high outlier ratio.

The 3D LiDAR SLAM has been widely deployed to mobile agents through different processing and optimization methods, e.g., iterative closest point (ICP) [2] method. Rather than directly using downsampled raw point clouds, local representative geometric features are applied to scan matching. A surface registration approach can achieve more refinement results by performing a point-to-plane tactic [3]. This was further generalized in [4] to perform ICP with a plane-to-plane strategy to enhance robustness. In order to reduce the impact of subtle environmental changes, normal distributions transform (NDT) [5] is tailored to scan matching based on the Gaussian model. The 3D point clouds are first subdivided into each cell and the probability of each measuring point is assigned a normal distribution. Therefore, no specific correspondences have to be verified. The piecewise continuous and differentiable probability density is deployed to estimate egomotion between the consecutive LiDAR frames.

The 3D LiDARs for robot perception generically produce large streaming volumes. However, there lack of sufficient computing resources to process these data in most real-time

1558-0644 © 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on March 31,2025 at 03:12:31 UTC from IEEE Xplore. Restrictions apply.

processing application scenarios. Therefore, the feature-based techniques have been extensively utilized [6]–[9]. For instance, in order to perform point-to-line and point-to-plane scan-matching tactics [10], edge and planar features are extracted from raw point cloud according to the smoothness of points in diverse laser beams. The image-based segmentation approach is then incorporated in feature extraction [11], [12]. Notably, this segmentation method is limited to sparse point clouds. Similar to [10], the optimization process has been decoupled into two steps to achieve lightweight. A tightly coupled LiDAR-inertial odometry framework is proposed based on the incremental smoothing and mapping (iSAM2) framework [13], [14]. Without simultaneously processing the measured values of IMU and LiDAR, this may lead to the deemphasis of the correlation between various sensors data and the loss of some constraint information of landmarks. Some particular feature extraction metrics attempt to dispose of the irregular and nonrepetitive scan patterns of the solid-state LiDAR [15] and the scan matching is performed similar to [10].

In addition, more attention has been devoted to surfel-based scan matching [16], [17]. The dense projective normal ICP is tailored to estimate the egomotion between the range image of the current frame and the surfel map [16]. A novel range image semantic segmentation network [18] is applied to filter possible dynamic objects [17]. Meanwhile, the semantic information is tailored for refining multiclass ICP registration and generating high-quality semantic maps. However, some structured 3D information is dissipated due to the procedure based on range image or scan line, and incorrect corresponding categorizes decrease the convergence of ICP nonlinear optimization. This leads to another increasing focus on learning-based LiDAR odometry. A novel scan-to-scan egomotion estimation network has been proposed in [19], and it can better interpret consecutive scans' spatial relations by checking normal consistency and locating authentic immobile regions using a learned uncertainly mask. A set of keypoints and corresponding feature descriptors are extracted from the raw point clouds [20], and the matching probabilities in all the dimensions of them are tailored to estimate the egomotion. However, the performance of these methods is limited by insufficient training data.

Segmentation of 3D point clouds is another indispensable step toward the situational assessment pipeline to understand the surrounding environments. A slope-robust cascaded ground segmentation approach is proposed to perform two principal steps [21]. By using the prior knowledge of LiDAR installation height and vertical resolution, the inter-ring distance-based filter is applied for primary nonground points. The RANdom SAmple Consensus (RANSAC) method [22] is then tailored to refine the precise ground points. Unfortunately, the calibration of mechanical LiDAR is laborious in practice. The Euclidean distance is employed to complete the classification of each scan line roughly [23]. Then, the adjacent points of various scan lines within the specified threshold are merged into the same category. Notwithstanding, the irregularity of the point cloud makes it tricky to distinguish the neighbor points of various beams. The innovative LiDAR-optimized curved voxel is deployed to segment the 3D point clouds [24], which can satisfy better discriminations by considering some critical clustering attributes. However, this approach fails to distinguish two adjacent objects when their distance is closer than the resolution of LiDAR. The random sampling module is integrated into a semantic segmentation network to reduce computational and memory costs [25]. In general, it is rather difficult to achieve precise segmentation in point clouds with various densities by using the constant Euclidean distance calculation method.

In this article, a series of amelioration has been implemented to address the deficiencies existing in the current LiDAR slam framework. The truncated least squares (TLS) method is applied innovatively for scan matching to make T-LOAM be more robust with outliers and alleviate its dependence on the initial value solution. The original point cloud is segmented by elaborately devising preprocessing module to extract four distinctive features: edge features, sphere features, planar features, and ground features. To materialize smoother odometry, three kinds of residual functions associated with disparate features are constructed by the TLS method. Finally, globally consistent maps are built as illustrated in Fig. 1.

Compared with other three state-of-the-art methods, F-LOAM[1], A-LOAM[2], and SuMa [16], the pose estimation error attributed to drift of T-LOAM has been evaluated on the KITTI odometry benchmark [1]. Experimental results have shown that T-LOAM achieves superior performances than others in most sequence experimental scenarios. More specifically, our contributions are listed as follows.

1) A novel, computationally efficient, and robust LiDAR-only odometry framework based on TLS is presented to accomplish robust navigation and construct a globally consistent map.
2) A series of implementations has been made to improve the segmentation precision of 3D LiDAR point clouds by utilizing multiregion ground extraction (MRGE) and dynamic curved-voxel clustering (DCVC).
3) The first LiDAR-based scheme developed through the Open3D [26] modern 3D data processing and released the source code of our implementation.

Our method is inspired by the graduated nonconvexity (GNC) algorithm [27], which has been proved efficient in executing various matching and optimization tasks in the computer vision field. We construct a novel pose optimization function based on TLS according to the convex optimization theory to estimate six-DoF egomotion in the consecutive frames of 3D LiDAR. The implementation follows the overview as shown in Fig. 2 and is set up for parallelization through multithreading and OpenMP [28] to guarantee the overall running efficiency of the algorithm.

Furthermore, robot operating system (ROS) nodelet package is exploited to process with zero-copy transport between messages of each node.

The remainder of this article is organized as follows. The hardware performed for experiments is introduced in

[1]https://github.com/wh200720041/floam.git
[2]https://github.com/HKUST-Aerial-Robotics/A-LOAM.git

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on March 31,2025 at 03:12:31 UTC from IEEE Xplore. Restrictions apply.

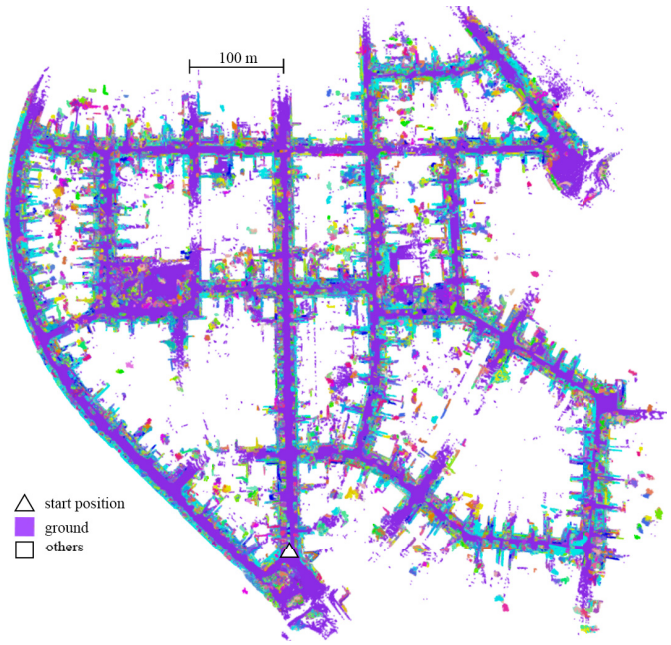

Fig. 1. Mapping result of T-LOAM on sequence 00 of KITTI odometry benchmark [1] with Velodyne HDL-64E, the total length of the sequence is 3724 m with 4541 frames of point clouds. The colors in the map are encoded with the segmentation results of the DCVC module.

Section II. An overview of the proposed system pipeline and the details of principal components are given in Sections III and IV, respectively, followed by experiments in Section V and conclusions in Section VI.

II. Datasets and System Hardware

The 3D LiDAR sensors are utilized for experimental validation, including Velodyne VLP-16 and HDL-64E, and both of them rely on mechanical spinning mechanisms to enable the 360° field of view (FoV). The egomotion coordinate system of position and orientation is employed in our framework, as shown in Fig. 3. The VLP-16 provides a vertical angular resolution of 2° to enable the 30° (±15°) vertical FoV and distance measurement over more than 100 m with a precision of 3 cm. The HDL-64E with the high vertical resolution (0.4°) is provided by the KITTI odometry benchmark [1], which possesses 26.9° vertical FoV.

The TX2 robot utilized in this article is a teleoperated vehicle, which is driven by Dual Brushless Motors and powered by a 670-Wh lithium battery. The LiDAR's installation height is set to be 0.75 m to guarantee omnidirectional scanning. The experimental data set is recorded by remote control. Also, the LiDAR sampling frequency is set at 10 Hz.

Two computers are adopted for assessing the real-time capability of the proposed framework, including an Nvidia Jetson TX2 and a laptop with a 2.5-GHz i7-10750H CPU. The Jetson TX2 is an embedded computing device equipped with an ARM Cortex-A57 CPU. All the modules are implemented by C++ and integrated into the ROS [29] with nodelet package as unique nodes in Ubuntu 18.04 Linux to accomplish zero-copy transport between them. The experiments presented in this article only utilize the CPUs in these systems.

III. Preprocessing Module

A. Framework Overview

The proposed T-LOAM framework is shown in Fig. 2. The 3D LiDARs are rotationally deskewed based on uniform motion. Then, the point clouds are fed into the MRGE module to obtain foreground $\mathbb{F}_k^v$ and background $\mathbb{F}_k^b$. After that, the DCVC method is tailored to segment $\mathbb{F}_k^v$ and filters out unstable categories to obtain $\mathbb{F}_k^d$. In addition, four distinctive feature points represented by $\mathbb{F}_k = \{F_k^e, F_k^s, F_k^p, F_k^g\}$ are processed by the feature extraction module. Through the preprocessed scans, the corresponding feature points are registered to the submap by the pose optimization module and a globally consistent map is constructed ulteriorly. The optimized pose of consecutive LiDAR scans is tailored to deskew the translational part of the current features, which will be merged into the submap for the next step.

B. Multiregion Ground Extraction

The ground points generically occupy a large proportion of the 3D point clouds recorded by autonomous vehicles and have simple mathematical models for processing convenience. More specifically, these features can be directly utilized to constrain pose components. However, merely using a single plane model is insufficient for an accurate representation of distribution in complex terrain. Therefore, we employ the MRGE approach to enhance segmentation precision.

In Algorithm 1, the raw point clouds from LiDAR scanning are first divided into various quadrants (default 4) according to the polar diameter and azimuth angle, as shown in Fig. 4(a). Each quadrant will be split into several equal subregions (default 3) ulteriorly, as shown in Fig. 4(b). The boundary of each subregion can be calculated by

$$\theta_i = \theta_s + \frac{n}{b}\alpha \cdot k_i \tag{1}$$

Algorithm 1 MRGE

Input: 3D point clouds G; number of iterations η; distance threshold τ; number of quadrants q; number of regions m;

$\mathbb{F}_k^v = \mathbb{F}_k^b = \emptyset$;
for Q_i in $Q_{1:q}$ **do**:
  $\{S_1, S_2, \ldots, S_m\} \leftarrow Q_i$;
  for S_r in $S_{1:m}$ **do**:
    S_g = **ExtractSeedPoints**(S_r, τ);
    for $t = 1$ to η **do**
      model = **EstimateBestPlane**($\mathbb{F}_k^b$);
      if model($p_i \in S_k$) $< \tau$ **then**
        $\mathbb{F}_k^b \leftarrow p_i$;
      else
        $\mathbb{F}_v \leftarrow p_i$;
      end
    end
  end
end
EstimateBestPlane according to (5).

Output: foreground $\mathbb{F}_k^v$, background $\mathbb{F}_k^b$.

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on March 31,2025 at 03:12:31 UTC from IEEE Xplore. Restrictions apply.

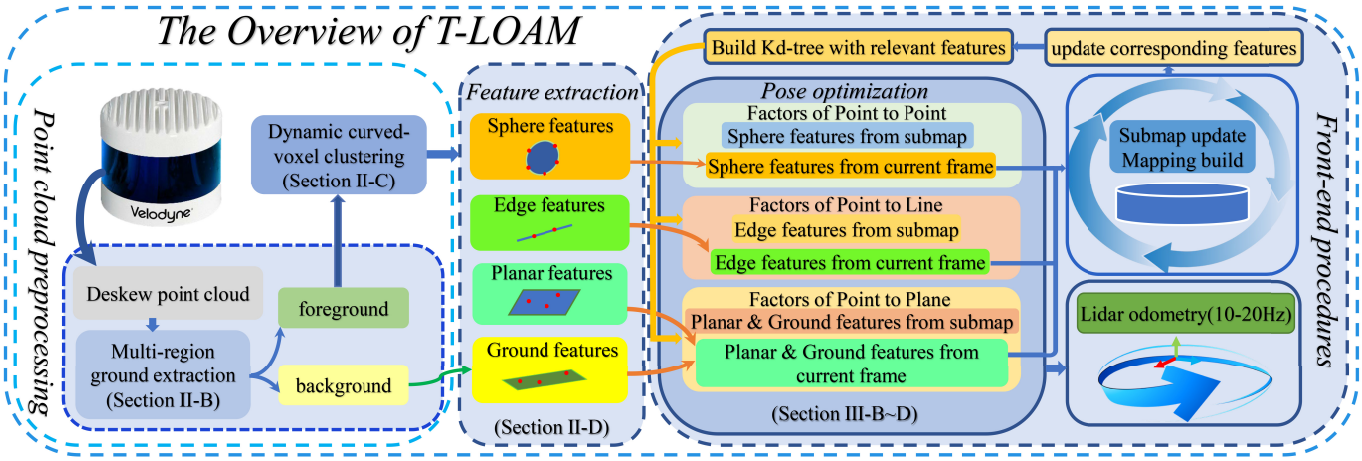

Fig. 2. Overview of the proposed T-LOAM system. Each frame of the 3D LiDAR is processed as input. Four main processing modules are introduced to construct the backbone of the algorithm. (a) MRGE module. (b) DCVC module. (c) Feature extraction module. (d) Pose optimization module. The particulars of these modules are discussed in Sections II and III.

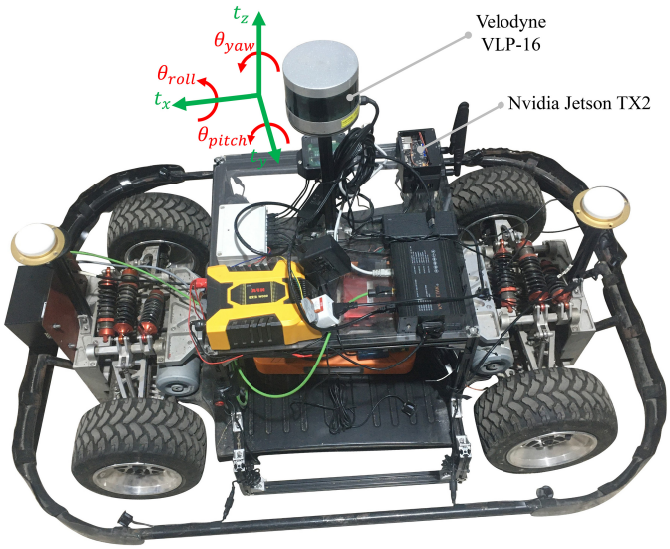

Fig. 3. TX2 robot experimental platform is equipped with Velodyne VLP-16 and Nvidia Jetson TX2 with ARM Cortex-A57 CPU. The installation height of the LiDAR is 0.75 m, which guarantees the scanning of the ground and space at the same time.

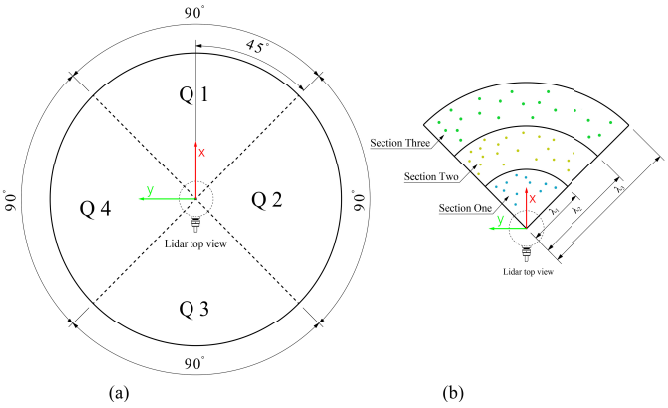

Fig. 4. (a) Various quadrants distribution diagram (default value is 4). (b) Distribution diagram of different regions in each quadrant (default value is 3), and various color points represent the ground seeds in the subregion.

$$\lambda_i = \begin{cases} h\left(\dfrac{1}{\tan(\theta_i)}\right), & \text{if } i \geq 1 \\ 0, & \text{if } i = 0 \end{cases} \tag{2}$$

where θ_s indicates the starting pitch angle of LiDAR, b and n represent the number of subregions and the sum of scan line, in which the beams may contain the ground points, respectively, α signifies the vertical resolution of the LiDAR, h indicates the installation height of the LiDAR, and k and i denote the regional coefficient and index, respectively.

For autonomous vehicles, the height coordinate component of the ground points is located at the lowest position generically. Therefore, *a priori* knowledge can be exploited to sort out the region clouds according to the height value. Seed points are selected from them within the specified threshold τ and are primarily utilized to fit the initial plane model. In order to increase the model accuracy, the multiaxis linear regression method is tailored to calculate relevant parameters and the principal directions are weighted to alleviate the impact of outliers. A linear plane model is tailored to reflect the distribution of the subregion followed by:

$$\begin{aligned} ax + by + cz + d &= 0 \\ n^T \mathrm{p} &= -d \end{aligned} \tag{3}$$

where $n = [a\ b\ c]^T$ and $\mathrm{p} = [x\ y\ z]^T$. The normal n can be reckoned according to (6) through the points of each subregion.

The covariance matrix M is utilized to capture the dispersion of the designated seed points in each subregion represented by

$$\mathcal{M} = \sum_{i=1}^{|s|} (s_i - \overline{s})(s_i - \overline{s})^T = \begin{pmatrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \\ c_1 & c_2 & c_3 \end{pmatrix} \tag{4}$$

where $\overline{s} \in \mathbb{R}^3$ denotes the mean of all points $s_i \in S$ in the subregion.

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on March 31,2025 at 03:12:31 UTC from IEEE Xplore. Restrictions apply.

Next, the three principal direction vectors of the specified set are calculated by

$$v_x = \begin{bmatrix} b_2c_3 - b_3b_3 \\ a_3b_3 - a_2c_3 \\ a_2b_3 - a_3b_2 \end{bmatrix}, \quad v_y = \begin{bmatrix} a_3b_3 - a_2c_3 \\ a_1c_3 - a_3a_3 \\ a_2a_3 - a_1b_3 \end{bmatrix},$$
$$v_z = \begin{bmatrix} a_2b_3 - a_3b_2 \\ a_2a_3 - a_1b_3 \\ a_1b_2 - a_2a_2 \end{bmatrix}. \tag{5}$$

In order to mitigate the effects of outliers on the prediction of normal vectors, each principal direction is weighted, and linear regression is applied to refine the normal vector. The weight value is the two norms of each principal direction illustrated by

$$n = \sum_{k\in\{x,y,z\}} w_k v_k$$
$$w_x = v_x\ [0]^2, \quad w_y = v_y\ [1]^2, \quad w_z = v_z\ [2]^2 \tag{6}$$

where operator [] represents the component value of the vector.

The parameters d can be evaluated based on (3) by superseding p with $\overline{s}$, which gives a preferable reflection of the point belonging to the final plane model when the normal n is obtained.

C. Dynamic Curved-Voxel Clustering

For accurately and efficiently segmenting 3D point clouds from the LiDAR sensor, a novel DCVC approach is proposed. The improved type of spatial voxel satisfies the three important aspects described in [24], as well as being more correspond with the point clouds' spatial distribution.

1) Definition 1 (Dynamic Curved Voxel): The ith, jth, and kth dynamic curved voxel indicates a 3D spatial voxel unit, which can be calculated by the configuration information of LiDAR presented as

$$DCV_{i,j,k} = \{P(\rho,\theta,\phi) = |\rho_i \le \rho < \rho_i + \Delta\rho_i, \theta_j \le \theta < \theta_j + \Delta\theta_j, \phi_k \le \phi < \phi_k + \Delta\phi_k\} \tag{7}$$

where each $P(\rho,\theta,\phi)$ is in polar coordinate with the radial distance ρ, azimuth angle θ, and polar angle ϕ.$\Delta\rho_i$, $\Delta\theta_j$, and $\Delta\phi_k$ indicate the increment of each voxel unit adjusted according to the sparsity and distance value of the point clouds.

In addition, the increment values of voxels in each direction can be calculated by

$$\Delta\rho_i = \rho_s - (a \times s_i + b) \times \Delta\rho$$
$$\Delta\theta_j = s_j \times \Delta\theta$$
$$\Delta\phi_k = s_k \times \Delta\phi \tag{8}$$

where ρ_s represents the starting voxel polar diameter, s is the step size, $\Delta\rho$, $\Delta\theta$, and $\Delta\phi$ indicate the increment of polar diameter, polar, and azimuth angle, respectively, and a and b are voxel coefficients which can be determined according to the point clouds of various beams and density of 3D LiDAR.

In order to better understand the practical distribution, two adjacent spatial dynamic voxels are visualized in Fig. 5. It is shown that three critical aspects for segmenting the 3D point clouds algorithm, including distance from the sensor, directional resolutions, and rarity of points, can be adequately satisfied. The sparsity coefficient can be utilized to adjust the volume of voxels respecting distinct distances in the polar radial direction, which avoids failing to distinguish the adjacent objects in space. The DCVC method is summarized in Algorithm 2. We first transform the rectangular coordinates of the point cloud into polar coordinates to establish dynamic curved voxel simultaneously. Then, the nonempty dynamic voxels are mapped into the hash table in order to enhance searching efficiency. After that, we search the nearest voxels surrounding the target voxel of the current point according to the hash-table mapping relationship and merge them into the same label. Finally, we return each point's category information and provide synthetic judgment to filter out the tiny groups and potential dynamic objects to obtain the final point clouds $\mathbb{F}_k^d$. In this way, the occurrence of incorrect segmentation classifications is significantly restrained when various objects remain adjacent to each other.

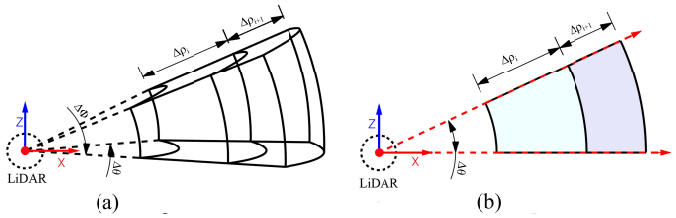

Fig. 5. (a) Perspective view of two adjacent dynamic curved voxel where $\Delta\rho$, $\Delta\phi$, and $\Delta\theta$ indicate unit voxel sizes of each polar coordinate. (b) Top view of two consecutive voxels of various sizes (in azure and lavender) and two contiguous vertical laser beams emitted from the LiDAR (in red). Each curved voxel has multiple edges parallel to the laser beam.

Algorithm 2 DCVC

Input: Foreground $\mathbb{F}_k^v$; $\rho_s, a, b, \Delta\rho, \Delta\theta, \Delta\phi$;
$\mathbb{F}_k^v \leftarrow$Convert to polar coordinate($\mathbb{F}_k^v$);
Build dynamic curved-voxels with $\rho_s, a, b, \Delta\rho, \Delta\theta, \Delta\phi$;
Map non-empty voxel into hash-table;
for every point p_i in $\mathbb{F}_k^v$ **do**
  neigh-index$\leftarrow$**FindNeighbors**(p_i, hash-table);
  labels$\leftarrow$**MergeLabels**(p_i, neigh-index);
  if p_i has no label information **then**
    labels$\leftarrow$**AddNewLabelInfo**(p_i, labels, neigh-index);
  end
end
$\mathbb{F}_k^{\mathrm{d}} \leftarrow$**LabelStatisticsAndFiltering**(labels);
Output: $\mathbb{F}_k^{\mathrm{d}}$ DCVC segmentation results;

D. Features Extraction

It is evident that the algorithms presented in [10] are prone to deterioration in geometric degradation scenarios, which will directly attenuate the precision of the LiDAR odometry or even make it fail. To enhance the stability of LiDAR-only odometry,

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on March 31,2025 at 03:12:31 UTC from IEEE Xplore. Restrictions apply.

T-LOAM will extract four distinctive geometric features from $\mathbb{F}_k^d$ and $\mathbb{F}_k^b$, including edge features F_k^e, sphere features F_k^s, planar features F_k^p, and ground features F_k^g. The background $\mathbb{F}_k^b$ is first downsampled directly as F_k^g due to enough precision of the MRGE module. The edge features are roughly extracted based on the smoothness of the local surface similar to [10], which is formulated by

$$c = \frac{1}{|s| \cdot \|p_i\|} \left\| \sum_{m \in s, m \neq i} (p_m - p_i) \right\| \tag{9}$$

where s is the set of consecutive points of the same laser beam (default value is 10), which contains half of its neighbor points p_m on each side of p_i.

Furthermore, the sphere features F_k^e and planar features F_k^p of the vertical part can be obtained by the principal component analysis (PCA) algorithm. It is utilized to capture the dispersion of the local neighborhood's dispersion, including curvature value, primary and secondary direction, normal vector, and corresponding eigenvalues. For implementation, the covariance matrix M is first calculated by the following equations:

$$\mathcal{M} = \frac{1}{k} \sum_{i=1}^{k} (p_i - \overline{p_k})(p_i - \overline{p_k})^T \tag{10}$$

where k and $\overline{p_k}$ represent the total number and mean value of current neighborhood points, respectively (we set k to 10 in our implement).

Furthermore, we perform the singular value decomposition (SVD) decomposition on the covariance matrix M to obtain the eigenvalues and corresponding eigenvectors that are denoted as λ and υ, respectively. The various features can be distinguished through the covariance matrix eigenvectors, which are associated with the direction of the most significant data variance. Some dispersed points are filtered primarily according to approximate equality of the eigenvalues. Then, the flatness and sphericity [30] are formulated by

$$\gamma = \frac{\lambda_3}{\lambda_1 + \lambda_2 + \lambda_3} \tag{11}$$

$$\sigma = \frac{\lambda_2 - \lambda_3}{\lambda_1}, \quad \psi = \frac{\lambda_3}{\lambda_1} \tag{12}$$

where γ is the curvature value and σ and ψ indicate flatness and sphericity coefficient, respectively. Note that the eigenvalues are given in descending order.

As shown in Algorithm 3, if the flatness coefficient σ is larger than the planar feature threshold τ, then points are extracted as the vertical part of the planar features. For the nonplanar part, if the sphericity coefficient ψ is more prominent than sphere feature threshold π, then points are appended to the sphere features. The extraction schematic of different features from HDL-64E LiDAR is shown in Fig. 6. Note that the improved extraction algorithm (besides edge features) is radically different from [10] in the respect of computing local smoothness on each beam of LiDAR. Furthermore, the principal direction and normal vector will be further exploited to the residual function in order to refine the horizontal and vertical distributions of the corresponding features.

Algorithm 3 Feature Extraction

Input: point clouds $\mathbb{F}_k^{\mathrm{d}}, \mathbb{F}_k^b$; smoothness η; flatness τ; sphericity π; edge direction threshold μ; planar normal threshold υ;

```
F_k^e ←RoughEdgeExtraction(F_k^d, η);
CalculatePCAFeatures(F_k^d);
for every point p_i in F_k^d do
    if σ > π then
        F_k^s ← p_i
    else if ψ > π then
        F_k^p ← p_i
    end
end
F_k^e, F_k^p ←refineVerticalDistribution(F_k^e, F_k^p, μ, υ);
F_k^g ←Downsampling(F_k^b);
```

Output: $\mathrm{F}_k^e, \mathrm{F}_k^s, \mathrm{F}_k^p$, and F_k^g;

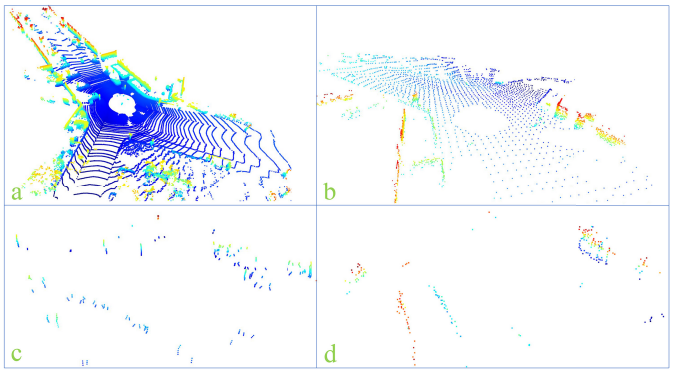

Fig. 6. Feature extraction process for a scan (000000.bin) on sequence 00 of the KITTI odometry benchmark. (a) Original point clouds. In (b), these points represent planar features included vertical and horizontal parts in F_k^p and F_k^g. In (c), the points indicate that edge features are mainly vertical distribution in F_k^e. In (d), these points are spherical features are uniformly distributed in F_k^s.

IV. Pose Optimization

Given the extracted edge, sphere, planar, and ground features, the pose optimization module is tailored to perform the scan-to-submap matching for estimating LiDAR measurements egomotion. Unfortunately, there are still inhomogeneous and irregular outliers in the various feature points, which leads to the deterioration of odometer precision in several ways. To overcome this problem, robust kernel functions are broadly deployed to the optimization process of most algorithms such as Huber loss [31] and lead to other nonconvexities to decrease the convergence rate simultaneously. These tactics have been employed by most of the frameworks. Furthermore, convex losses possess a low breakdown point and are still sensitive to gross outliers [32]. Inspired by GNC [27], the TLS approach is integrated into the pose optimization module to resist the outliers of each feature residual.

A. State Description and Update

The global frame and the LiDAR frame are denoted as w and ℓ, respectively. By the Lie algebra, the pending robot

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on March 31,2025 at 03:12:31 UTC from IEEE Xplore. Restrictions apply.

state x_k^w and corresponding tangent vector can be written as

$$x_k^{w\wedge} = \begin{bmatrix} [\theta]_\times & \rho \\ 0 & 0 \end{bmatrix} \in \mathfrak{se}\ (3), \quad x_k^{w\wedge} = [\rho, \theta]^{\mathrm{T}} \in \mathbb{R} \quad (13)$$

where $[\theta]_\times$ is a skew-symmetric matrix and ρ and θ denote the corresponding translational and rotational parts of vector elements, respectively.

Since manifolds are locally homeomorphic to $\mathbb{R}^n$, we utilize bijective Exp and Log to map the vector tangent of state $\mathbb{R}^6 \cong \mathfrak{se}(3)$ to 3D rigid motion group $T_\ell^w = [R|t] \in \mathrm{SE}(3)$ with $R \in SO(3)$ a rotation matrix and $t \in \mathbb{R}^3$ a translation vector, which represents the transformation from $\boldsymbol{\ell}$ to w, described by

$$T_\ell^w = \mathrm{Exp}(x_k^w) = \begin{bmatrix} \mathrm{Exp}(\theta) & J_l(\theta)\rho \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \in \mathbb{R}^{4\times 4} \quad (14)$$

$$x_k^w = \mathrm{Log}(\mathrm{T}_\ell^w) = \begin{bmatrix} J_l^{-1}(\theta)t \\ \mathrm{Log(R)} \end{bmatrix} \quad (15)$$

where $\mathrm{Exp}(x_k^w)$ is the exponential map, $\mathrm{Log}(R)$ is its inverse map, $J_l(\phi)$ is the left Jacobians of Lie groups, and $J_l^{-1}(\phi)$ is its inverse. Their specific form can be found in [33].

The robot state x_k^w is updated when the pose increment δx_k^w is obtained during the optimization process illustrated by

$$\tilde{x}_k^w = \mathrm{Log}\big(\mathrm{Exp}\big(x_k^w\big) \cdot \mathrm{Exp}\big(\delta x_k^w\big)\big). \quad (16)$$

B. General Form of TLS

Apart from considering the hypothesis of the Gaussian noise model [34], each feature noise is assumed to have an upper bound. Therefore, the noise value associated with each measurement is denoted explicitly as σ_i, and $\|\sigma_i\| \le \beta_i$, where β_i is the upper bound of the specified noise. The general form of TLS can be represented by

$$f(T) = \underset{T\in \mathrm{SE}(3)}{\arg\min} \frac{1}{N} \sum_{i=1}^{N} \min\left(\frac{1}{\beta_i^2}\|b_i - T \otimes a_i\|^2, \overline{c}^2\right) \quad (17)$$

where $a_i, b_i \in \mathbb{R}^3$ are the points of two 3D point clouds, T represents the transformation matrix of two sets, and $\overline{c}^2$ indicates the proportional coefficient, which can dispose of potential outliers in a rigorous or more tolerant way. More specifically, if $\overline{c}^2$ is set to 1, potential outliers are filtered out entirely and more iteration steps are required in the meanwhile.

The tiny residual $(1/\beta_i^2\|b_i - T \otimes a_i\|^2 \le \overline{c}^2)$ of LiDAR measurements will be manipulated to calculate the least-squares solution. When the measured value $(1/\beta_i^2\|b_i - T \otimes a_i\|^2 > \overline{c}^2)$ generates a sizeable residual value, it will be discarded and replaced by a permanent value without influencing the optimization result.

C. Residual of Point-to-Point

The spherical feature points $\mathrm{p}_k^\ell \in \mathbb{S}$ are extracted from the current sweep in the LiDAR frame $\boldsymbol{\ell}$. We evaluate point-to-point residuals considering the distribution of spherical features. The latest sphere features of n frames are preserved into the corresponding local feature submap $\mathbb{M}_s^w$. Furthermore, the relationship between the current LiDAR frames and submap is described as

$$\mathbb{M}_s^w = \overleftarrow{F}_i^s \cup \overleftarrow{F}_{i-1}^s \cup \ldots \cup \overleftarrow{F}_{i-n}^s \quad (18)$$

where $\overleftarrow{F}_i^s, \ldots, \overleftarrow{F}_{i-n}^s$ are the transformed sphere features in w using the transformations $\{T_{\ell i}^w, \ldots, T_{\ell i-n}^w\}$. In this article, n is chosen to be 3 for HDL-64E or 5 for VLP-16. We first search one nearest spherical point p_k^w in the local feature submap $\mathbb{M}_s^w$ about p_k^ℓ. Then, two corresponding feature points are constructed into point-to-point residual using the TLS method represented by

$$\mathcal{R}_s\left(x_k^w, \mathrm{p}_k^\ell, \mathbb{M}_s^w\right) = \mathrm{p}_k^w - T_\ell^w \otimes \mathrm{p}_k^\ell$$

$$f\left(x_k^w, \mathrm{p}_k^\ell, \mathbb{M}_s^w\right)_s = \underset{x_k^w \in \mathfrak{se}(3)}{\arg\min} \frac{1}{N} \sum_{k=1}^{N} \min\left(\frac{1}{\beta_i^2}\left\|\mathcal{R}_s^k\right\|^2, \overline{c}^2\right). \quad (19)$$

D. Residual of Point-to-Line

An edge feature $\mathrm{p}_k^\ell \in \mathbb{E}$ and its associated direction vector v_e are provided by the feature extraction module in $\boldsymbol{\ell}$. The latest n frames are stitched the corresponding local feature submap $\mathbb{M}_e^w$, and it will cut out the point cloud beyond the specified range to diminish the uncorrelated features simultaneously

$$\mathbb{M}_e^w = V_{xyz} \cap \left[\overleftarrow{F}_i^e \cup \overleftarrow{F}_{i-1}^e \cup \ldots \cup \overleftarrow{F}_{i-n}^e\right] \quad (20)$$

where $\overleftarrow{F}_i^e, \ldots, \overleftarrow{F}_{i-n}^e$ is the transformed edge features in w using the transformations $\{T_{\ell i}^w, \ldots, T_{\ell i-n}^w\}$ and V_{xyz} represents the clipped voxel range related to the translation of the current robot state component. Each direction is set to be 100 m, and n is the same as above.

In order to warrant that the current point p_k^ℓ is approximated by vertically rectilinear distribution, we search five adjacent points in $\mathbb{M}_e^w$ and calculate the mean $\overline{e}^w$ and covariance matrix. Then, it is decomposed by SVD to obtain the primary principal direction vector n_e. If the largest of eigenvalue is substantially larger than the rest and the z-axis component of n_e is greater than the given threshold value μ (see line 4 in Algorithm 3), neighborhood points are thought to be distributed in the vertically rectilinear distribution and refine two more representative points $\overrightarrow{e}^w = \overline{e}^w + \delta n_e$ and $\overleftarrow{e}^w = \overline{e}^w - \delta n_e$ belonging to the straight line. After that, it is constructed into the form of TLS by

$$\mathcal{R}_e\left(x_k^w, \mathrm{p}_k^\ell, \mathbb{M}_s^w\right) = \frac{\left|\left(\mathrm{p}_k^w - \overrightarrow{e}^w\right) \times \left(\mathrm{p}_k^w - \overleftarrow{e}^w\right)\right|}{\left|\overrightarrow{e}^w - \overleftarrow{e}^w\right|}$$

$$f\left(x_k^w, \mathrm{p}_k^\ell, \mathbb{M}_s^w\right)_e = \underset{x_k^w \in \mathfrak{se}(3)}{\arg\min} \frac{1}{N} \sum_{k=1}^{N} \min\left(\frac{1}{\beta_i^2}\left\|\mathcal{R}_e^k\right\|^2, \overline{c}^2\right). \quad (21)$$

E. Residual of Point-to-Plane

The feature point is denoted as $\mathrm{p}_k^\ell \in \mathbb{P}$ indicating planar distribution. The corresponding local ground feature submap $\mathbb{M}_g^w$ and the planar feature submap $\mathbb{M}_p^w$ are updated in the same way as we did for $\mathbb{M}_s^w$ and $\mathbb{M}_e^w$. We then transform them into its global frame w and search five planar feature points in the corresponding local frame submap $\mathbb{M}_g^w$ or $\mathbb{M}_p^w$, namely, $\mathbb{S}^w = \{s_i^w\} \in \mathbb{M}_g^w$ or $\mathbb{M}_p^w$, with $i = 1, \ldots, 5$. The same approach described in Section III-A is tailored to achieve the best plane model fitting using the adjacent points $\mathbb{S}^w$. We compute the distance of the current point p_k^ℓ to the plane model and append

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on March 31,2025 at 03:12:31 UTC from IEEE Xplore. Restrictions apply.

this residual to pose optimization within the threshold range, which can be formulated by

$$\begin{aligned}\mathcal{R}_p\left(x_k^w, \mathrm{p}_k^\ell, \mathbb{M}_p^w\right) &= n^T \otimes \left(T_\ell^w \otimes \mathrm{p}_k^\ell\right) + d \\ f\left(x_k^w, \mathrm{p}_k^\ell, \mathbb{M}_p^w\right)_p &= \underset{x_k^w \in \mathfrak{se}(3)}{\arg\min} \frac{1}{N} \sum_{k=1}^{N} \min\left(\frac{1}{\beta_i^2}\left\|\mathcal{R}_p^k\right\|^2, \overline{c}^2\right).\end{aligned} \tag{22}$$

F. Iterative Pose Optimization

Due to the nonlinearity of the residual function and the strong nonconvexity of the Lie groups SE(3), the TLS problem cannot be solved trivially. By means of the black-Rangarajan duality [35] and GNC [27], the primal problem can otherwise be transformed to

$$\begin{aligned} f(x_k^w) &= \underset{x_k^w \in \mathfrak{se}(3), w_k \in [0,1]}{\arg\min} \frac{1}{N} \sum_{k=1}^{N} \left[w_k \mathcal{R}^2\left(x_k^w, \mathrm{p}_k^\ell\right) + \Phi_\mu(w_k)\right] \\ \Phi_\mu &= \frac{\mu(1-w_k)}{\mu + w_k} \overline{c}^2 \end{aligned} \tag{23}$$

where $w_k \in [0,1] (k = 1, \ldots, N)$ are slack variables (or weights) that quantify the contribution of each feature residual, μ indicates the control parameter, and the function $\Phi_\mu(w_k)$ denotes the penalty term of the weight value by GNC (the so-called outlier handing process).

Note that GNC is favorable for solving most nonconvex cost functions. Pivotal ideas and examples of interpretation are detailed in [27]. We originate from its convex surrogate and gradually changing μ (i.e., generically increasing the amount of nonconvexity) until the original nonconvex function is retrieved. In each iteration, the weight w_k associated with each feature residual is primarily fixed and the pending optimized state variables x_k^w are resolved. The fixed x_k^w is tailored to optimize over the corresponding weight w_k for the next step. More specifically, the following process is performed at each inner iteration t.

1) Pose Variable Update: Minimize (24) with respect to $x_k^{w(t)}$ with fixed weights $w_k^{(t-1)}$

$$x_k^{w(t)} = \underset{x_k^w \in \mathfrak{se}(3)}{\arg\min} \frac{1}{N} \sum_{k=1}^{N} w_k^{(t-1)} \mathcal{R}^2\left(\widetilde{x}_k^w, \mathrm{p}_k^\ell\right) \tag{24}$$

where the second term in (24) has nothing to do with pose variables $x_k^{w(t)}$, which can be removed directly. Then, (25) is simplified weighted least squares, which can be resolved using the Gauss–Newton method.

2) Weight Update: Minimize (25) with respect to $w_k^{(t)}$ with fixed $x_k^{w(t)}$

$$w_k^{(t)} = \underset{w_k \in [0,1]}{\arg\min} \frac{1}{N} \sum_{k=1}^{N} \left[w_k^{(t-1)} \mathcal{R}^2\left(x_k^{w(t)}, \mathrm{p}_k^\ell\right) + \Phi_\mu\left(w_k^{(t-1)}\right)\right] \tag{25}$$

where $\mathcal{R}^2(x_k^{w(t)}, \mathrm{p}_k^\ell)$ is a constant for fixed $w_k^{(t)}$, denoted as $\overline{\mathcal{R}_k}^2$. The residual error derivative concerning the weight value in (25) can be directly retrieved from the closed form at iteration t, which is described as (27).

The update process of the control parameter μ is carried out as follows. It originates from the substitutive convex set and then progresses the value of μ in each iteration until the original cost is recovered. After the pose variable is updated, the closed-form solution can be resolved to assign the corresponding weight $w_k^{(t)}$ by

$$w_k^{(t)} = \begin{cases} 1 & \text{if } \overline{\mathcal{R}_k}^2 \in \left[0, \ \dfrac{\mu}{\mu+1}\overline{c}^2\right] \\ \dfrac{\overline{c}}{\overline{\mathcal{R}_k}^2}\sqrt{\mu(\mu+1)} - \mu & \text{if } \overline{\mathcal{R}_k}^2 \in \left[\dfrac{\mu}{\mu+1}\overline{c}^2, \ \dfrac{\mu+1}{\mu}\overline{c}^2\right] \\ 0 & \text{if } \overline{\mathcal{R}_k}^2 \in \left[\dfrac{\mu+1}{\mu}\overline{c}^2, \ +\infty\right]. \end{cases} \tag{26}$$

In fact, after the pending pose variable x_k^w is refreshed for the first time, we reinitialize μ according to the maximum residual amidst various feature residuals by

$$\mu = \frac{\overline{c}^2}{\left(2\mathcal{R}_{\max}^{(t)2} - \overline{c}^2\right)}, \quad \mathcal{R}_{\max}^{(t)} = \max\left\{\mathcal{R}_e^{(t)}, \mathcal{R}_s^{(t)}, \mathcal{R}_g^{(t)}, \mathcal{R}_p^{(t)}\right\} \tag{27}$$

where $\mathcal{R}_e^{(t)}, \mathcal{R}_s^{(t)}, \mathcal{R}_g^{(t)}$, and $\mathcal{R}_p^{(t)}$ symbolize edge, sphere, ground, and planar feature residual, respectively.

The control parameter μ is increased at each iteration by $\mu = \exp^{\eta * t}$ in order to accelerate the convergence speed and decrease the operating time (note that η is the increment factor). The overall objective function incorporates four components are described as

$$\begin{aligned} f\left(x_k^w\right) = f\left(x_k^w, \mathrm{p}_k^\ell, \mathbb{M}_e^w\right)_e + f\left(x_k^w, \mathrm{p}_k^\ell, \mathbb{M}_s^w\right)_s \\ + f\left(x_k^w, \mathrm{p}_k^\ell, \mathbb{M}_g^w\right)_g + f\left(x_k^w, \mathrm{p}_k^\ell, \mathbb{M}_p^w\right)_p. \end{aligned} \tag{28}$$

Finally, the process of iterative pose optimization is shown in Algorithm 4. The corresponding Jacobian matrix is appended to the Ceres Solver [36] and the nonlinear optimization problem in (25) is solved. The iteration is terminated after the quantity of the weighted feature residuals converges at each outer iteration. Note that the Jacobian matrices of all residual functions will eventually follow the right disturbance model of Lie algebras. Moreover, the multithread technique is tailored to assemble the residual function of each feature during each iteration in order to improve computational performance (see lines 4–7 in Algorithm 4).

V. Experimental Validations

In this section, the performance of each module in our algorithm is verified exhaustively through experiments on the KITTI odometry benchmark [1], which provides point clouds collected from the Velodyne HDL-64E S2 at a typical frequency of 10 Hz. The data set is collected from various street scenarios ranging from the inner city to highway traffic. It is noteworthy that the highway data sets lack geometric features in the vertical direction, and some places, e.g., a long corridor scenario, are lack forwarding constraints. Other fast moving dynamic objects are also included almost all the time. Due to additional sphere features, the stability of the

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on March 31,2025 at 03:12:31 UTC from IEEE Xplore. Restrictions apply.

Algorithm 4 Pose Optimization

Input : local submap $\mathbb{M}^w = \{\mathbb{M}_e^w, \mathbb{M}_s^w, \mathbb{M}_g^w, \mathbb{M}_p^w\}$; current frame features $\mathbb{F}_k = \{\mathrm{F}_k^e, \mathrm{F}_k^s, \mathrm{F}_k^g, \mathrm{F}_k^p\}$; number of iterations n; noise bound β ; control parameter μ; increment factor η; residuals and weights $\{\mathcal{R}_e, \mathcal{R}_s, \mathcal{R}_g, \mathcal{R}_p, w_e, w_s, w_g, w_p\}$;

Initialize parameters $\mathrm{x}_k^w, \mu, \mathcal{R}_e, w_e, \mathcal{R}_s, w_s, \mathcal{R}_g, w_g, \mathcal{R}_p, w_p$;

for $t = 1$ to n **do**

  // parallel addition for CPU with multiple cores line 4-7

  AddEdgeCostFactor($\mathrm{F}_k^e, \mathbb{M}_e^w, w_e, \mathcal{R}_e$);

  AddSphereCostFactor($\mathrm{F}_k^s, \mathbb{M}_s^w, w_s, \mathcal{R}_s$);

  AddSurfCostFactor($\mathrm{F}_k^g, \mathbb{M}_g^w, w_g, \mathcal{R}_g$);

  AddGroundCostFactor($\mathrm{F}_k^p, \mathbb{M}_p^w, w_p, \mathcal{R}_p$);

  $\mathrm{x}_k^w \leftarrow$**CeresSolve();**

  reinitialize μ at $t = 1$ according to (28);

  if *the nonlinear optimization converges* **then**

  break;

  $\mu \leftarrow \mu * \exp^{\eta * t}$;

end

Output: pose increment between consecutive LiDAR frames.

proposed algorithm has been improved in the highway scenario. Simultaneously, the scheme is further verified in the campus scenarios recorded by the TX2 robot with VLP-16 (see Fig. 3) in order to assure adequate applicability and practical application preference. Eventually, the performance of each module is justified with respect to the computational time at the representative sequences.

A. Validation of MRGE and DCVC

The performances of MRGE and DCVC approaches are verified on the KITTI odometry and campus data sets. The three persuasive experimental scenarios, including intersections and crossroads provided by the HDL-64E, another specific sparse scenario gathered from the VLP-16, are considered with the experimental results shown in Figs. 7 and 8. The ground points with purple marks represent the MRGE module. By splitting the raw frame into multiple regions and the most suitable plane model fitting, it is conceivable to correct the erroneous category for the ground points. The distance of the dynamic spatial voxel generically decreases with the range of the 3D LiDARs, thereby rectifying the variance of point cloud sparsity. Based on these improved adaptations, it can be seen that our technique delivers precise measures of numerous objects in dense scenarios with adjacent vehicles accurately segmented in Fig. 7. It is ulteriorly verified by segmenting two adjacent persons in a sparsity scenario shown in Fig. 8.

B. Validation on the KITTI Odometry Dataset

The precision of T-LOAM is primarily evaluated on the KITTI odometry benchmark [1]. The competitors include the following state-of-the-art LiDAR-based odometry frameworks:

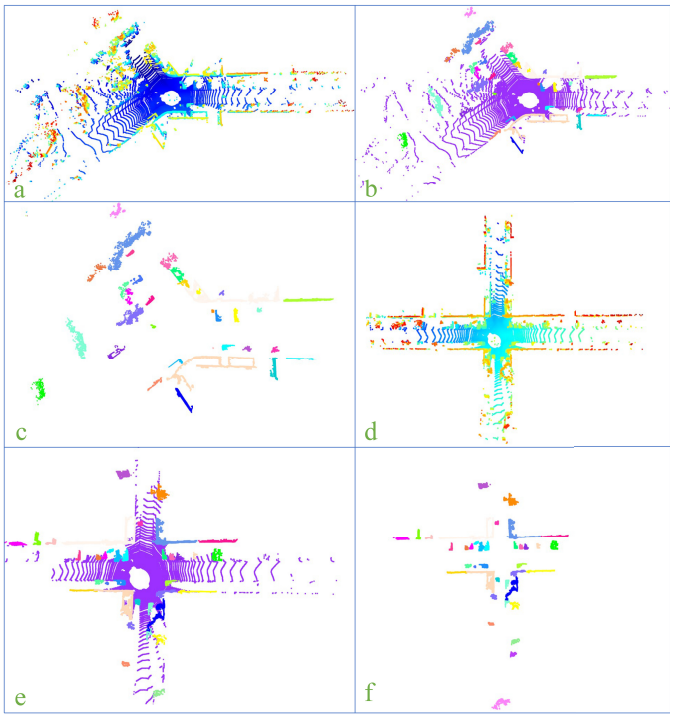

Fig. 7. DCVC segmentation results on the KITTI sequence 00 odometry benchmark. (a)–(c) Segmentation result of the intersection scenario (000000.bin), where (a) is the raw point cloud. The purple clouds in (b) represent the ground points obtained by MRGE. (c) Segmentation result by DCVC, and different colors represent various categories. (d)–(f) Another crossroad scene with more vehicles (000110.bin).

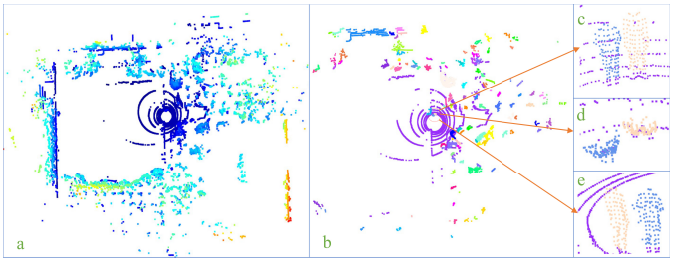

Fig. 8. DCVC segmentation results on the VLP-16 LiDAR collected by TX2 in campus scenarios. (a) Raw point cloud. (b) Classification results of various objects. The circle marks position denotes two adjacent persons after segmented by DCVC. More detailed visualization results are shown in (c) front view, (d) top view of (d), and (e) back view.

F-LOAM[1], A-LOAM[2], and SuMa [16]. The evaluation results of all sequences are presented in Table I. Concerning pure odometry without corrections through loop closures, it can be concluded that our method can achieve superior precision than others, especially in the highway situations when lacking geometric features in the perpendicular orientation. It forward constrains like a long corridor (see Seq01 in Table I). Furthermore, T-LOAM can generate smoother odometry trajectories and develop globally consistent maps (see Seq 00 in KITTI odometry benchmark). The comparison with the ground truth, which is collected based on differential GPS, is shown in Fig. 9.

The variations of the proposed scheme position and orientation on the KITTI odometry sequence 00 benchmark [1] are

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on March 31,2025 at 03:12:31 UTC from IEEE Xplore. Restrictions apply.

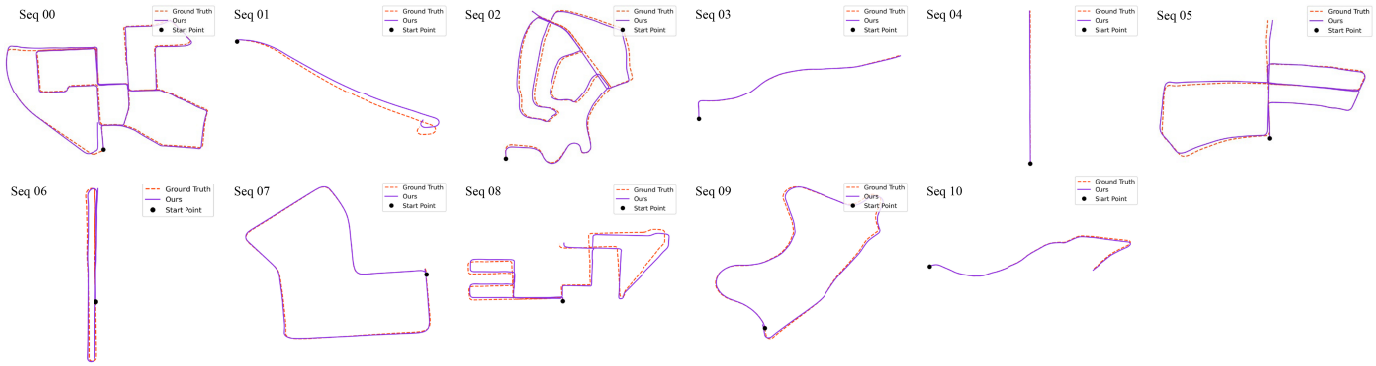

Fig. 9. Trajectory diagram concerning the training set and the corresponding sequence error results are shown in Table I. The orange-red trajectory is collected based on differential GPS ground truth that can accomplish the localization precision is around cm level. The optimized odometry of our scheme is presented in the blue-violet trajectory, which has been aligned with the ground truth. T-LOAM has dramatically improved under degradation scenarios, such as Sequences 03, 06, and 10.

TABLE I
RESULTS OF EVALUATION IN THE KITTI ODOMETRY DATASET

Sequence	00	01	02	03	04	05	06	07	08	09	10	Average
Ours	**0.98**/0.60	**2.09**/**0.52**	**1.01**/**0.39**	**1.10**/0.82	**0.68**/0.68	**0.55**/**0.32**	**0.56**/**0.31**	**0.50**/0.47	**0.94**/**0.33**	**0.80**/**0.40**	**1.12**/**0.61**	**0.93**/**0.49**
F-LOAM[1]	1.11/0.40	3.01/0.85	1.22/0.43	4.51/1.84	0.93/0.63	0.63/0.32	2.15/0.74	0.51/0.35	0.97/0.37	0.82/0.40	2.52/0.96	1.67/0.66
A-LOAM[2]	4.82/1.90	3.90/1.08	5.50/2.40	5.50/2.41	1.51/1.15	4.80/1.92	3.10/1.12	2.94/1.79	4.77/2.08	5.73/1.86	3.48/1.39	4.18/1.73
SuMa[16]	2.10/0.90	4.00/1.20	2.30/0.80	1.40/0.70	11.9/1.10	1.50/0.80	1.00/0.60	1.80/1.20	2.50/1.00	1.90/0.80	1.80/1.00	2.92/0.91

The evaluation results of the T-LOAM in the KITTI odometry dataset(training) using the KITTI evaluation odometry tool which the relevant trajectory error length is the mean value of 100-800 length: relative translational error in % / relative error in degrees per 100m. Note that pure odometry is evaluated without corrections at loop closures in all algorithms.

shown in Fig. 10. The localization error of translation and orientation is significantly alleviated by T-LOAM; especially, the 2D pose components (x, z, yaw) are approachable to the precision of ground truth. Note that the positional and orientation error of T-LOAM chiefly proceeds from the convergence precision of altitude, roll, and pitch, which has a trivial impact on the flat terrain navigation task. This is the prevailing problem in most LiDAR-only odometry schemes, caused by the fact that it is arduous to precisely capture changes in slope and instantaneous orientation only through 3D point clouds.

Although loop detection can help correcting the drift, our current work mainly concentrates on improving the precision of the frontend odometry. In practice, erroneous loop closure may occur at homologous scenarios or when the agone position has not returned for a long time. It should also be mentioned that authentic improvement of loop closure can hardly be verified via the KITTI odometry benchmark due to some inconsistencies in the height of the ground truth recorded by the GPS-based inertial navigation system.

Finally, the position and orientation error has been evaluated on the test sets using the parameters mentioned above. It is shown that T-LOAM yields a less average translational error of 0.93% and an average rotational error of 0.0049°/m comparing to 1.67% translational error and 0.0066°/m rotational error of the F-LOAM on the KITTI odometry benchmark.

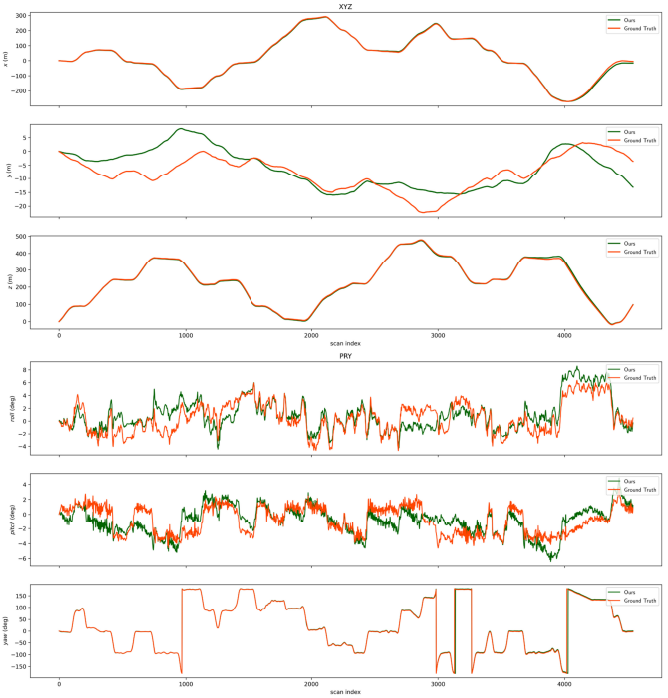

Fig. 10. Change of position and orientation contrasted with ground truth on the KITTI sequence 00 odometry benchmark. Note that the result has been converted to the KITTI Point Gray Flea 2 Video Cameras coordinate system through calibration (xyz → zxy).

C. Validation on Campus Environment

The proposed T-LOAM scheme is further deployed on the TX2 robot with VLP-16 3D LiDAR in the Wuhan University of Technology campus environments with numerous buildings, trees, roads, and sidewalks.

1) Experiment I: The primary experiment is designed to demonstrate that T-LOAM can accomplish low-drift

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on March 31,2025 at 03:12:31 UTC from IEEE Xplore. Restrictions apply.

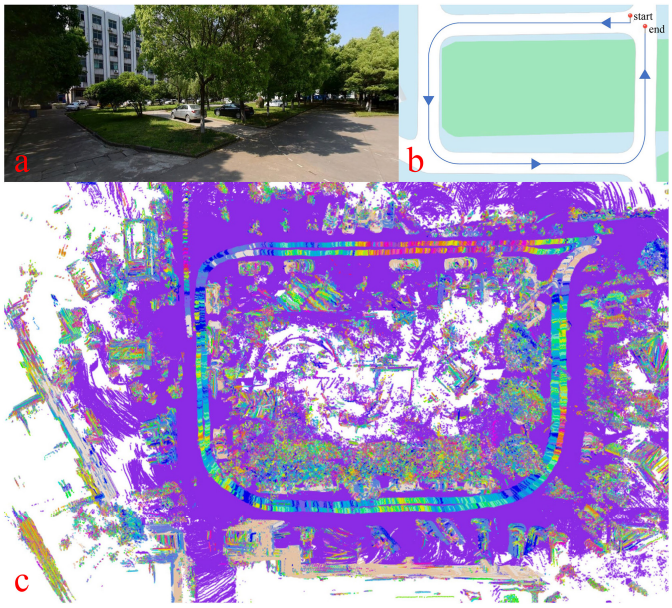

Fig. 11. Evaluation results on experiment I. (a) and (b) represent the environment and topological map provided by Google maps, respectively. Blue trajectory and arrow direction in (b) indicate the TX2 robot path. 3D map of experiment I from T-LOAM is shown in (c). The colors are rendered through the DCVC module.

egomotion and the reestablished map having global consistency. To avoid violent yaw maneuvers, the TX2 robot is controlled to move forward smoothly in the direction indicated by the path arrow in the topology map, as shown in Fig. 11(b). Here, the LiDAR sampling frequency remains 10 Hz. The distribution of various features is relatively uniform and can be obtained in each consecutive frame of VLP-16. The starting and ending positions of the robot are marked in Fig. 11(b). We feed on initial translational and rotational guess based on the assumption of uniform motion in order to improve the convergence speed of the algorithm. The robot returns to the approximately same position to record a total of 1400 frames of 3D point clouds with an average speed of 1.25 m/s. The globally consistent map is tailored to reconstruct by T-LOAM, as shown in Fig. 11(c).

2) Experiment II: The following experiment is developed to validate further the stability of the proposed scheme in more complex scenarios subject to deterioration. The TX2 robot is propelled forward for a long trajectory according to the path in the topological map, as shown in Fig. 12(b), and again, aggressive yaw maneuvers are avoided. As a result, only few stable features can be utilized for registration when passing through some areas in this scenario. Many road sections do not possess available planar features, and the distribution of trees is also relatively sparse. The low vertical resolution of the VLP-16 LiDAR gives rise to more sparse point clouds of objects beyond 10 m, which brings additional challenges simultaneously. Only some edge features can be extracted from tree trunks and lamppost for scan matching. In order to improve stability and avoid losing forward constraints, the proportion of spherical features is increased accordingly. Some unstable environmental characteristics are filtered out by the DCVC module. More specifically, the approximate

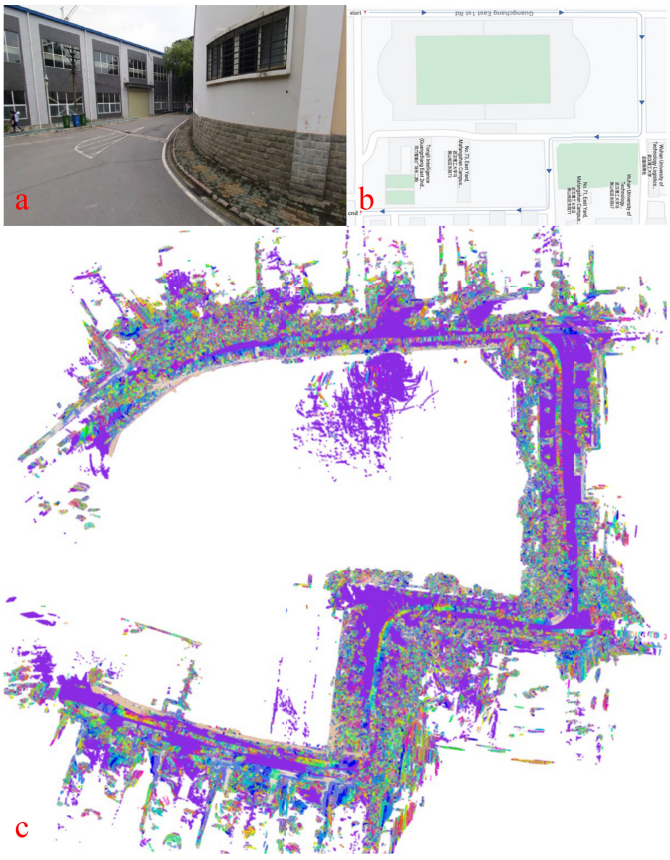

Fig. 12. Experimental results of scenes with relatively degenerate features that the topology provided by Google maps. The environment and the TX2 robot driving path are shown in (a) and (b), respectively. The globally consistent map is indicated by (c). The purple color represents the ground point clouds.

proportion of various features is diminished to 20% F_k^e, 50% F_k^s, 5% F_k^g, and 25% F_k^p. The robot accumulates almost 3700 frames of 3D point clouds by the VLP-16 LiDAR after 15 min of driving at an average speed of 1.35 m/s. As shown in Fig. 12(c), it can be concluded that T-LOAM can not only precisely estimate the LiDAR egomotion under the circumstance of treacherous features distribution but also construct a globally consistent map by only using the 3D LiDARs sensor.

D. Validation of Runtime Performance

T-LOAM can show excellent real-time performance for all the above tests. In the preprocessing stage of our scheme, there are three running nodes, including MRGE, DCVC, and LiDAR odometry, which integrates feature extraction. As shown in Fig. 13, the average runtime of each processing node is recorded for representative sequences 00 on the KITTI odometry benchmark [1] (for a visual impression of the complexity, see also Fig. 1). All the computations are conducted by a laptop (Intel Core i7-1075 0H CPU, 8-GB RAM).

The preprocessing module mainly performs the segmentation of the 3D LiDAR point clouds to filter out unstable categories and extract ground features. Both of them are implemented with the multithread technique to improve the runtime performance. In the LiDAR odometry node, four

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on March 31,2025 at 03:12:31 UTC from IEEE Xplore. Restrictions apply.

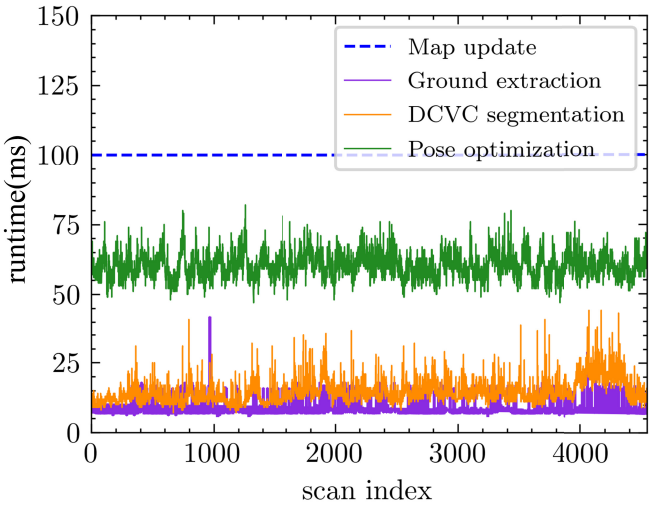

Fig. 13. Processing time of each module per frame on the KITTI sequence 00 odometry benchmark (see Fig. 1 for the finally registered complete point cloud).

various features are utilized for the optimization module in parallel to estimate egomotion. All the processing nodes are integrated into ROS nodelet package to process with zero-copy transport between them. Finally, the average runtimes of running MRGE node, DCVC node, and pose optimization node are 8, 14, and 60 ms, respectively. For all sequences recorded by diverse LiDARs (including Velodyne VLP-16 and HDL-64E) in numerous scenarios, the proposed framework is proved to be eligible for real-time performance assessment.

VI. Conclusion

This article presented T-LOAM, a novel optimization technique for real-time LiDAR-only odometry and mapping in various scenarios. The MRGE, DCVC, and feature extraction are developed as processing modules of the scheme, which provides a good tradeoff between performance indicators, such as precision and overhead cost. The proposed method is validated on both the KITTI odometry benchmark and campus environment. The experimental results have shown that our proposed T-LOAM can achieve real-time capability and delivers superior segmentation and mapping precision over the state-of-the-art LiDAR odometry frameworks, especially in the feature-degrading scenarios.

Other navigation sensors will be integrated for future work to accomplish the robust navigation tasks of the T-LOAM ulteriorly in more specific challenging scenarios. Besides, we will exploit a place recognition method to guarantee that loop closure can be detected precisely when too many drifts in odometry estimation occur.

References

[1] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, 2013. [Online]. Available: http://www.cvlibs.net/datasets/kitti/eval_odometry.php

[2] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat, "Comparing ICP variants on real-world data sets," *Auton. Robots*, vol. 34, no. 3, pp. 133–148, Apr. 2013.

[3] Y. Chen and G. Medioni, "Object modelling by registration of multiple range images," *Image Vis. Comput.*, vol. 10, no. 3, pp. 145–155, Apr. 1992.

[4] A. Segal, D. Haehnel, and S. Thrun, "Generalized-ICP," in *Proc. Robot., Sci. Syst. (RSS)*, vol. 2, no. 4, Seattle, WA, USA, 2009, p. 435.

[5] P. Biber and W. Straßer, "The normal distributions transform: A new approach to laser scan matching," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, vol. 3, Oct. 2003, pp. 2743–2748.

[6] R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu, "Fast 3D recognition and pose using the viewpoint feature histogram," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2010, pp. 2155–2162.

[7] Y. Li and E. B. Olson, "Structure tensors for general purpose LiDAR feature extraction," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2011, pp. 1869–1874.

[8] 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.

[9] 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. Automat. (ICRA)*, May 2020, pp. 2095–2101.

[10] J. Zhang and S. Singh, "LOAM: LiDAR odometry and mapping in real-time," in *Proc. Robot., Sci. Syst. (RSS)*, vol. 2, no. 9, 2014.

[11] 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.

[12] I. Bogoslavskyi and C. Stachniss, "Fast range image-based segmentation of sparse 3D laser scans for online operation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2016, pp. 163–169.

[13] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "LIO-SAM: Tightly-coupled lidar inertial odometry via smoothing and mapping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Las Vegas, NV, USA, Oct. 2020, pp. 4758–4765.

[14] 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.

[15] J. Lin and F. Zhang, "Loam livox: A fast, robust, high-precision LiDAR odometry and mapping package for LiDARs of small FoV," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2020, pp. 3126–3131.

[16] J. Behley and C. Stachniss, "Efficient surfel-based SLAM using 3D laser range data in urban environments," in *Proc. Robot., Sci. Syst. (RSS)*, vol. 2018, 2018.

[17] X. Chen, A. Milioto, E. Palazzolo, P. Giguere, 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.

[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] Q. Li *et al.*, "LO-Net: Deep real-time LiDAR odometry," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 8473–8482.

[20] W. Lu, Y. Zhou, G. Wan, S. Hou, and S. Song, "L3-Net: Towards learning based LiDAR localization for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 6389–6398.

[21] P. Narksri, E. Takeuchi, Y. Ninomiya, Y. Morales, N. Akai, and N. Kawaguchi, "A slope-robust cascaded ground segmentation in 3D point cloud for autonomous vehicles," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2018, pp. 497–504.

[22] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1981. [Online]. Available: http://doi.acm.org/10.1145/358669.358692

[23] D. Zermas, I. Izzat, and N. Papanikolopoulos, "Fast segmentation of 3D point clouds: A paradigm on LiDAR data for autonomous vehicle applications," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2017, pp. 5067–5073.

[24] S. Park, S. Wang, H. Lim, and U. Kang, "Curved-voxel clustering for accurate segmentation of 3D LiDAR point clouds with real-time performance," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 6459–6464.

[25] Q. Hu *et al.*, "RandLA-Net: Efficient semantic segmentation of large-scale point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11108–11117.

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on March 31,2025 at 03:12:31 UTC from IEEE Xplore. Restrictions apply.

[26] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3D: A modern library for 3D data processing," 2018, *arXiv:1801.09847*. [Online]. Available: http://arxiv.org/abs/1801.09847 and http://www.open3d.org/
[27] H. Yang, P. Antonante, V. Tzoumas, and L. Carlone, "Graduated non-convexity for robust spatial perception: From non-minimal solvers to global outlier rejection," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 1127–1134, Apr. 2020.
[28] L. Dagum and R. Menon, "OpenMP: An industry standard API for shared-memory programming," *IEEE Comput. Sci. Eng.*, vol. 5, no. 1, pp. 46–55, Jan. 1998.
[29] M. Quigley *et al.*, "ROS: An open-source robot operating system," in *Proc. ICRA Workshop Open Source Softw.*, Kobe, Japan, 2009, vol. 3, no. 3, p. 5.
[30] T. Hackel, J. D. Wegner, and K. Schindler, "Fast semantic segmentation of 3D point clouds with strongly varying density," *ISPRS Ann. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. 3, pp. 177–184, Jun. 2016.
[31] L. Carlone and G. C. Calafiore, "Convex relaxations for pose graph optimization with outliers," *IEEE Robot. Autom. Lett.*, vol. 3, no. 2, pp. 1160–1167, Apr. 2018.
[32] P.-Y. Lajoie, S. Hu, G. Beltrame, and L. Carlone, "Modeling perceptual aliasing in SLAM via discrete-continuous graphical models," *IEEE Robot. Autom. Lett.*, vol. 4, no. 2, pp. 1232–1239, Apr. 2019.
[33] J. Solà, J. Deray, and D. Atchuthan, "A micro lie theory for state estimation in robotics," 2018, *arXiv:1812.01537*. [Online]. Available: http://arxiv.org/abs/1812.01537
[34] M. Milanese, "Estimation and prediction in the presence of unknown but bounded uncertainty: A survey," in *Robustness in Identification and Control*. Boston, MA, USA: Springer, 1989, pp. 3–24.
[35] M. J. Black and A. Rangarajan, "On the unification of line processes, outlier rejection, and robust statistics with applications in early vision," *Int. J. Comput. Vis.*, vol. 19, no. 1, pp. 57–91, Jul. 1996.
[36] S. Agarwal and K. Mierle. (2012). *Ceres Solver*. [Online]. Available: http://ceres-solver.org/

Pengwei Zhou received the B.S. degree in vehicle engineering from the Hubei University of Automotive Technology (HUAT), Shiyan, China, in 2017. He is pursuing the master's degree with the AIPS Laboratory and the Key Laboratory of Automotive Intelligent Perception and Safety, Wuhan University of Technology, Wuhan, China, under the supervision of Dr. X. Guo.

His research interests include light detection and ranging (LiDAR) slam and multisensor fusion.

Xuexun Guo received the B.E. degree from the Department of Mechanical Engineering, Huazhong University of Science and Technology, Wuhan, China, in 1982, and the Ph.D. degree in mechanical engineering from the Beijing Institute of Technology, Beijing, China, in 1995.

He has been a Professor with the Wuhan University of Technology, Wuhan, since 2001. His research fields are automobile design, power drive systems, and simulation and test of electrical vehicles.

Xiaofei Pei was born in Wuhan, Hubei, China, in 1985. He received the B.S. degree in mechanical engineering from the Wuhan University of Technology, Wuhan, China, in 2007, and the Ph.D. degree in mechanical engineering from the Beijing Institute of Technology, Beijing, China, in 2013.

From 2011 to 2012, he was a Visiting Student with Rutgers University, New Brunswick, NJ, USA. He has been an Associate Professor with the Wuhan University of Technology since 2016. His research interests include vehicle active safety and autonomous vehicles.

Ci Chen was born in Wuhan, Hubei, China, in 1984. He received the B.S. degree in electronic engineering from the Wuhan University of Technology, Wuhan, China, in 2007, and the Ph.D. degree in electronic engineering from Dublin City University, Dublin, Ireland, in 2015.

He has been a Lecturer with the Wuhan University of Technology since 2015. His research interests include vehicle control and optimal control.

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on March 31,2025 at 03:12:31 UTC from IEEE Xplore. Restrictions apply.