IEEE Access
Multidisciplinary ⁝ Rapid Review ⁝ Open Access Journal

Received 13 January 2023, accepted 30 January 2023, date of publication 2 February 2023, date of current version 9 February 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3241800

APPLIED RESEARCH

A Lightweight Real-Time 3D LiDAR SLAM for Autonomous Vehicles in Large-Scale Urban Environment

GUANGMAN LU[1], **HONG YANG**[2], **JIE LI**[1], **ZHENFEI KUANG**[1], **AND RU YANG**[2]
[1]School of Electronics and Communication Engineering, Guangzhou University, Guangzhou 510006, China
[2]School of Mechanical and Electrical Engineering, Guangzhou University, Guangzhou 510006, China

Corresponding author: Hong Yang (yanghong@gzhu.edu.cn)

This work was supported in part by the Guangdong Provincial Natural Science Foundation Team Project of under Grant 2017B030312001.

ABSTRACT For autonomous vehicles, real-time localization and mapping in the unknown environment is very important. In this paper, a fast and lightweight 3D LiDAR simultaneously localization and mapping (SLAM) is presented for the localization of autonomous vehicles in large-scale urban environments. A novel encoding approach based on depth information is proposed to encode unordered point clouds with various resolutions, which avoids missing dimensional information in the projection of point clouds onto a 2D plane. Principal components analysis (PCA) is modified by dynamically selecting neighborhood points according to the encoded depth information to fit the local plane with less time consuming. The threshold and the number of feature points are adaptive according to distance intervals, results in sparse feature points extracted and uniformly distributed in the three-dimensional space. The extracted significant feature points improve the accuracy of the odometer and speed up the alignment of the point cloud. The effectiveness and robustness of the proposed algorithm are verified on the KITTI odometry benchmark and MVSECD. A fast runtime of 21 ms is obtained for the odometer estimation. Compared to several typical state-of-the-art methods on the KITTI odometry benchmark, the proposed approach reduces translation errors by at least 19% and rotation errors by 7.1%.

INDEX TERMS 3D SLAM, localization, autonomous driving, loop closure, large-scale urban area.

I. INTRODUCTION

As one of core techniques in autonomous driving [1], localization and navigation of vehicles have been a research hotspot. The algorithm needs to process a large amount of data [2], and the lightweight algorithm is especially important. These technologies are of great value to the automobile industry [3]. Localization and navigation generally can be realized by high-definition maps and GPS. However, high-definition maps and GPS are not always available in all urban scenarios, such as out of service induced by tunnels, overpasses, shielding of tall buildings, and technical problems. The localization in the absence of high-definition maps and GPS can be addressed by the ego-motion estimation of SLAM. Stable technical algorithms provide security for autonomous driving [4]. SLAM is capable of solving position estimation problems of vehicles in unknown environment, and constructs the map of the surrounding environment. It generally can be classified by visual SLAM [5], [6], [7] and LiDAR SLAM [8], [9], [10], [11], [12] according to sensors (camera, LiDAR) employed. Passively acquiring information from cameras, visual SLAM is greatly affected by the illumination and texture conditions of surrounding environment. In contrast, 3D LiDAR SLAM relies on the active measurement of the geometric information of environment by LiDAR rather than texture information. Therefore, LiDAR SLAM is more robust and widely used in outdoor scenarios for practical applications. A complete LiDAR SLAM system includes a pose estimation odometer in the front-end and a global pose graph optimization in the back-end. The point cloud registration is the core in the front-end, from which the vehicle pose can be obtained. Typical approaches include Iterative

The associate editor coordinating the review of this manuscript and approving it for publication was Mohammad Alshabi.

Closest Point (ICP) [13], Normalized Distribution Transform (NDT) [14] and feature-based approaches. The classic ICP pairs the nearest points on the basis of Euclidean distance, and constantly optimizes the point-to-point correspondence in an iterative manner to obtain the transformation of the vehicle. However, the transformation of each point pair in a frame point cloud requires a large amount of computation. The NDT divides the point cloud into many small grids and then calculates the probability distribution between grids. The continuous differentiable probability density is employed to estimate the ego-motion between continuous LiDAR frames. However, the algorithm is sensitive to the rotation angle and the registration accuracy is affected by the grid size. Feature-based approaches only extract point clouds with features for matching to achieve excellent real-time performance together with high precision. However, the feature extraction approach is still a challenge for autonomous vehicles in urban scenes where geometric features are not obvious.

Based on the above approaches, several 3D LiDAR SLAM methods in outdoor have been proposed. LOAM [8] proposed point-to-line and point-to-plane registration of point clouds based on feature points to achieve an accurate odometer. However, it produced significant drift in the case of few geometric features for large-scale environments due to the lack of back-end optimization. LeGO-LOAM [9] proposed a fast and lightweight SLAM approach utilizing ground optimization, but the absence of ground point clouds brought more errors in the odometer. HDL_GRAPH_SLAM [10] is a 3D LiDAR SLAM framework based on graph optimization with loopback, which can fuse IMU, GPS and road constraints, etc., but the scan-to-scan point cloud registration method of front-end odometer is not accurate enough. LIOM [11] proposed a low drift and robust LiDAR inertial odometer and mapping method in large-scale fast-moving environment, but it has a large memory occupation and poor time performance. These solutions still remain drawbacks in large-scale urban environments.

In this paper, we focus on robust real-time localization and mapping of autonomous vehicles in complex large-scale urban environments. The number of the input point clouds is cut down by the removal of ground points based on plane fitting. Then the preprocessed unordered point clouds are encoded according to the depth information rather than being projected onto 2D plane. A robust dual-adaptive feature extraction algorithm based on the encoded depth information is proposed by modifying PCA. The pose of the vehicle is estimated by registering the extracted edge and plane points. Finally, a two-step loop detection is used to optimize the global map with eliminated cumulative error. The effectiveness and robustness of our system has been verified on the datasets KITTI [15] and MVSECD [16]. The main contributions of this paper are as follows:

1) Unordered point clouds are encoded using the depth information, which avoids the missing of dimensional information induced by the projection of point clouds onto 2D plane. The encoding is adaptive to kinds of resolution of point clouds without ordering point clouds by layers.
2) The approach of selecting neighborhood points in PCA is improved to dynamically select points according to the encoded depth information to fit the local plane, which can effectively reduce the time cost of the algorithm compared with the original PCA of selecting fixed points.
3) An approach is proposed for adaptively selecting the threshold and number of feature points in different distance intervals. This approach extracts more uniform sparse feature points in three-dimensional space than the traditional fixed number feature extraction method, which improves the odometer accuracy and reduces the time cost.

II. RELATED WORK

In 3D LiDAR SLAM, pose estimation is mainly accomplished by registering point clouds of two frames. Common point cloud registration in SLAM includes ICP [13] direct registration and feature point registration. ICP does not need to sort the input point cloud and only needs to find the nearest point to register, however it takes a lot of time to register each point. The method based on feature points needs to order the input disordered point cloud, however it can register quickly. The classical ICP iteration solves the point-to-point transformation. In order to improve the speed and accuracy of point cloud registration, the distance from registered point to plane is used as the error measure to pair points with local planes. GICP [17] modified the loss model by the matching between two local planes by combining the point-to-plane ICP and plane-to-plane ICP approaches. IMLS-SLAM [18] proposed an implicit moving least squares-based point-to-model alignment method that aligns sampled points with an implicit surface by extracting observable sampling points. These works improve ICP to some extent, however the real-time performance is still affected when large amounts of point cloud data are processed. Unlike ICP method, the feature point method only registers a small number of extracted feature points, thus registration is fast. LOAM [8] is a classical outdoor 3D LiDAR SLAM, which is the first to propose a method based on feature point registration. The input point clouds are ordered according to the angular resolution of the lidar. LOAM calculates the smoothness of several neighborhood point clouds on the same line beam to extract edge features and plane features, and optimizes the point-to-line and point-to-plane distance residuals to obtain the odometer poses. On the basis of LOAM, LeGO-LOAM [9] proposed a lightweight SLAM system. It ordered input disordered point clouds by projecting point clouds as distance images. Then, a two-step L-M nonlinear optimization was carried out on the segmented ground points and edge points, and the transformation coefficients of the two sets of point clouds were sequentially solved. However, projecting a point cloud onto 2D plane will lose one dimension information. Lio-sam [19] is a fusion method based on feature points that combines IMU

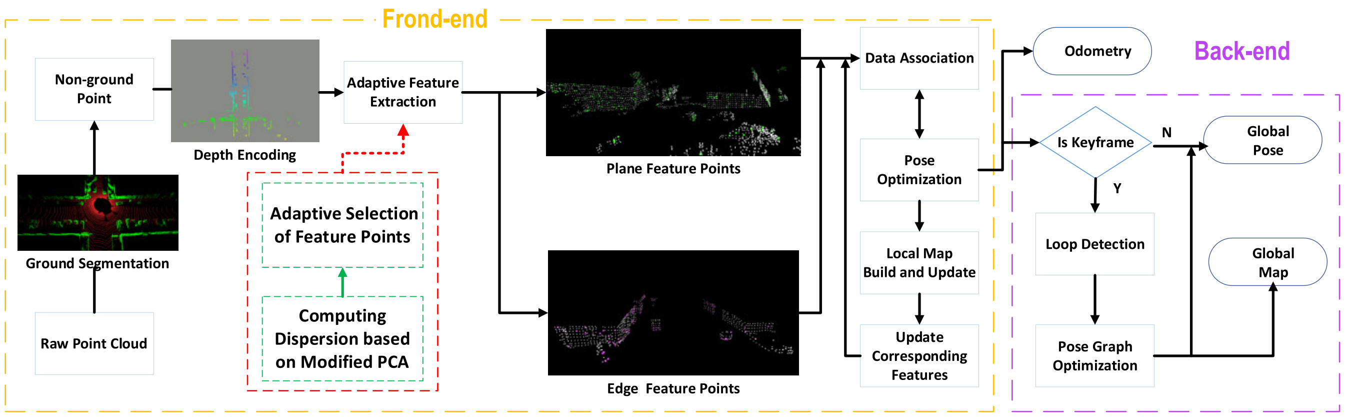


FIGURE 1. **The overview of the proposed 3D-SLAM system.**

and GPS factors, but it required joint calibration of sensors. F-loam [20] extracts the same number of feature points at different distances as several works above. It employed a non-iterative two-stage distortion compensation method to replace the iterative distortion compensation method, providing high computational efficiency and accurate pose. MULLS [21] proposed multi-metric linear least square iterative closest points algorithm based on the classified feature points. Principal components analysis (PCA) approach is employed to extract six fine features, including ground points, facade, roof pillar, beam and vertex. The feature points were classified for point-to-point, point-to-line and point-to-plane ICP registration to obtain a low drift and multifunctional pure LiDAR SLAM system.

III. FRAMEWORK OVERVIEW

The framework of the system proposed in this paper is shown in Figure 1, where the front-end acquires point cloud data from the sensors and pre-processes the raw point cloud to segment ground points. Non-ground points are ordered using depth information. Edge and plane features are extracted from the non-ground points by adaptive extraction method. The relative poses of the vehicle movements are obtained based on the alignment of the feature points in two consecutive frames. The odometer of the vehicle can be estimated by accumulating relative positions over time. The back-end receives the position from the odometer and judges that whether the vehicle has reached its previous position. Finally, a graph-based optimization approach is used to eliminate errors in the matching process to obtain globally consistent trajectories and mapping.

IV. METHODOLOGY

A. GROUND SEGMENTATION

Ground points generically occupy a large proportion of 3D point clouds recorded by autonomous vehicles. The number of point clouds can be reduced by ground segmentation on the input point clouds. The traditional ground segmentation methods include RANSAC [22] and the ray ground filter method [23]. RANSAC estimates model parameters by a random sample of observed data. Ray Ground Filter algorithm calculates the change of point radius on the same angle to acquire ground points. However, the above algorithms randomly select points from the whole point cloud, leading to slow runtime and segmentation errors. In this paper, we employ a fast ground filtering method [24], which selects seed point set as the prior value to speed up the algorithm. Firstly, a frame of point cloud is divided into n segments along the moving direction of the vehicle. The area in the x-axis direction is divided into a number of sub planes. Multiple sub planes are merged into one plane to reduce segmentation errors brought by the change of ground slope. Then, we select points with small values along the z-axis and calculate their average height H_m. The term th_z is the threshold to select seed point. A point is selected as a seed point set $S \in \mathbb{R}^3$ when its z value is less than the threshold $H_m + th_z$. The seed points are taken as the prior of the initial plane to speed up the plane fitting. The set of seed points is fitted into the initial plane. The cross-projection distance between each point and the initial plane is calculated. As shown in Figure 2 (a), if the distance is smaller than the artificial threshold th_g, the point will be taken as the ground point.

The mathematical expression used to estimate the plane model is as follows:

$$ax + by + cz + d = 0. \tag{1}$$

By expressing it in vector form:

$$\begin{bmatrix} a & b & c \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = -d,$$
$$\mathbf{n}^T \mathbf{p} = -d \tag{2}$$

$\mathbf{n}$ is solved by the covariance $M \in \mathbb{R}^{3\times3}$ and M describes the dispersion of the seed point set of each sub region:

$$M = \sum_{i=1}^{|S|} (s_i - \bar{s})(s_i - \bar{s})^T = U\Sigma V^T, \tag{3}$$

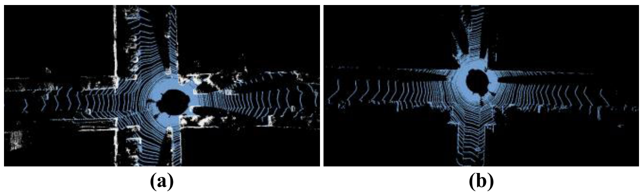


FIGURE 2. Point cloud ground segmentation. (a) Ground segmentation detection results. (b) Ground Point Cloud.

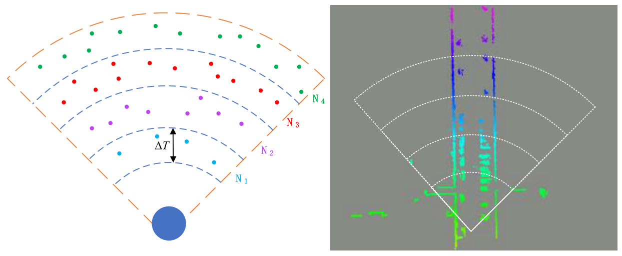


FIGURE 3. Depth encoding on point cloud.

where $\bar{s} \in \mathbb{R}^3$ is the mean of all points $s_i \in \mathbb{R}^3$, $U = M^T M$, $U \in \mathbb{R}^{3\times3}$, $V = MM^T$, $V \in \mathbb{R}^{3\times3}$ and $\Sigma = \mathrm{diag}(\sigma_1, \sigma_2, \sigma_3)$. The normal $\mathbf{n}$ is perpendicular to the plane, which indicates the direction with the least variance. The parameter d can be solved by replacing $\mathbf{p}$ in equation (1) with $\bar{s}$. The vertical distance between each point and the initial plane is calculated to judge whether it belongs to the next plane by comparing with the threshold Th_{dist} .Finally, all ground points after classification are regarded as the seed point set in the next iteration for iterative optimization. The merged ground point cloud by the sub-planes using the above algorithm is shown in Figure 2 (b).

B. POINT CLOUD ORDERED ENCODING

The input data acquired from LiDAR are usually unordered 3D point clouds, which can be transformed into organized point cloud sequence by projecting onto a 2D plane or classifying according angular information. Here, the depth information is adopted to organize the unordered input point cloud avoiding the loss of dimensional information or dependence on the resolution of LiDAR. The 3D scan is divided into N_r rings by equal intervals in the radial direction of the sensor coordinates. The index of rings is calculated as follows:

$$i = \begin{cases} \dfrac{D_j}{\Delta T}, & \textit{if integer} \\ \left[\dfrac{D_j}{\Delta T}\right] + 1, & \textit{otherwise}, \end{cases} \tag{4}$$

where $[\cdot]$ is an integer symbol, ΔT is the radial gap distance between the different rings, D_j is the depth information of each non-ground point cloud $p_j^N(x_j, y_j, z_j)$ that calculated by $D_j = \sqrt{x_j^2 + y_j^2 + z_j^2}$. When the interval is too large, the feature points are distributed sparsely, which affects the accuracy of odometer. In this paper, we use $\Delta T = 4$ m and $N_r = 21$. According to the distance value D_j of each point, the point p_j^N is divided into corresponding rings N_j (as shown in Figure 3). The point clouds subset of each ring is represented as follows:

$$\rho = \bigcup\nolimits_{i\,\in[N_r]} \rho_i, \tag{5}$$

where ρ_i is the set of points belonging to the *ith* ring, $[N_r] = \{1, 2, \ldots, N_{r-1}, N_r\}$.

After the point cloud is partitioned, each ring $R(i)$ is represented by a point cloud subset as:

$$R(i) = \rho_i. \tag{6}$$

Thus, the point clouds are classified to be sets with different distance indices D_{id}, and further processing on the point clouds are performed according to the index.

C. COMPUTING DISPERSION BASED ON MODIFIED PCA

Singular value points in the non-ground point cloud have adverse effect on the localization accuracy of the odometer due to no matching points during registering. Therefore, Singular value points need to be removed before feature extraction. Euclidean clustering method is employed to classify objects by clustering non-ground points. Outliers are classified and removed when the number of clustered point clouds is less than the threshold. The removal of outliers before feature extraction can reduce redundantpoints and increase the feasibility of feature points.

LOAM [8] proposed a feature extraction method to calculate the local smoothness of several neighborhood points on the same beam. However, it may fail in geometric degradation scenarios due to equal smoothness values. In order to improve the robustness of feature extraction algorithm, principal components analysis (PCA) algorithm is improved to extract point cloud features. In the original PCA algorithm, a fixed number of neighborhood points is used to fit the local plane. We improved the method by adaptively selecting neighborhood points according to distance intervals to fit the local plane. The number of neighborhood points k is defined as: $k = \alpha - [\beta D_{id}(i)]$, where $[\cdot]$ is an integer symbol, α and β are linear parameters. To reduce the searching computational cost, point clouds are stored in 3D KD-trees. The local domain dispersion is calculated by using these neighborhood points. The distance between the selected neighborhood point $p_i(x_i, y_i, z_i)$ and the local center is:

$$\partial_i = p_i - \frac{1}{k}\sum_{j=1}^{k} p_j = \left[\partial_{ix}, \partial_{iy}, \partial_{iz}\right], \tag{7}$$

where k is the total number of neighborhood points and $i = 1, 2, 3, \ldots, k$. Then the covariance matrix of the following k points are calculated by:

$$C = \frac{1}{k-1}\begin{bmatrix} \sum_{i=1}^{k} \partial_{ix}^2 & \sum_{i=1}^{k} \left(\partial_{ix}\partial_{iy}\right) & \sum_{i=1}^{k} \left(\partial_{ix}\partial_{iz}\right) \\ \sum_{i=1}^{k} \left(\partial_{ix}\partial_{iy}\right) & \sum_{i=1}^{k} \partial_{iy}^2 & \sum_{i=1}^{k} \left(\partial_{iy}\partial_{iz}\right) \\ \sum_{i=1}^{k} \left(\partial_{ix}\partial_{iz}\right) & \sum_{i=1}^{k} \left(\partial_{iy}\partial_{iz}\right) & \sum_{i=1}^{k} \partial_{iz}^2 \end{bmatrix} \tag{8}$$

The eigenvalues λ_i $(i = 1, 2, 3)$ and the corresponding eigenvectors $\mathbb{N}_i$ $(i = 1, 2, 3)$ of the covariance matrix C are determined by singular value decomposition (SVD). The eigenvalues are sorted to obtain $\lambda_1 > \lambda_2 > \lambda_3$. The eigenvector $\mathbb{N}_1$ corresponding to the maximum eigenvalue λ_1 is the primary vector, and the $\mathbb{N}_3$ corresponding to the λ_3 is the normal vector. These eigenvalues and eigenvectors can be used to extract edge and plane features of local point clouds. According to the relationship between eigenvalues, local linearity and planarity can be defined as [25]:

$$\delta = \frac{\lambda_1 - \lambda_2}{\lambda_1}, \tag{9}$$

$$\varepsilon = \frac{\lambda_2 - \lambda_3}{\lambda_1}, \tag{10}$$

where δ is the planarity and ε is the linearity. The feature points are selected based on the values of δ and ε. The modified PCA algorithm speeds up the feature extraction compared to the original PCA method. The feature extraction using adaptive selection of neighborhood points $(k = \alpha - [\beta D_{id}(i)], \alpha = 10, \beta = 0.1)$ is 4 ms faster than the fixed points $(k = 5)$.

D. ADAPTIVE SELECTION OF FEATURE POINTS

Generally, the selection threshold of feature points is fixed, together with the number of feature points (For example, [8] selects 20 edge points). However, point clouds obtained by LiDAR scanning are sparse at long distance and dense at short distance, resulting the nonuniform distribution of feature points. The nonuniform distribution has adverse effect on the accuracy of the odometer and the stability of the SLAM system. Therefore, we propose an adaptive selection method of feature points based on distance. According to Section 3.3, the index D_{id} of each point cloud is obtained in the encoding stage. The threshold χ of feature point selection is adaptively calculated according to the distance intervals as:

$$\chi = 1 - \gamma_t R_k, \quad t = (edge,\ plane), \tag{11}$$

where is k the distance number of each point, R_k is the range with the distance number k obtained via equation (4), γ is the linear factor. In general, the value of γ_{plane} is set larger than γ_{edge} due to more plane points needed by the pose estimation. As shown in Figure 4, the edge p_ε and plane p_S feature points are selected by comparing the obtained threshold and local dispersion values according to Equations (9) and (10):

$$\mathbb{P} = \begin{cases} p_S \leftarrow p^N & , if\, \delta > \chi_p \\ p_\varepsilon \leftarrow p^N & , if\, \varepsilon > \chi_l. \end{cases} \tag{12}$$

Different number of feature points are dynamically selected in different distance intervals. The adaptive number of feature points is mathematically expressed as follows: $num = \omega_t R_k$, $\omega < 1$, num is an integer. As shown in Figure 5, the feature points obtained by the adaptive selection method are sparser than those obtained by the fixed selection method. The extracted feature points are uniformly distributed in six degrees of freedom. The uniformly distributed feature points in the space of six dimensions brings constraints on each degree of freedom, and improves the accuracy of the odometer and the stability of the SLAM system. The complete process of feature extraction algorithm is shown in algorithm 1.

Algorithm 1 Feature Extraction

Parameters: p^N non-ground point cloud; p current point; δ/ε planarity / linearity value; χ_e/χ_pedge/ plane point selection threshold; N_e/N_pnumber of edge/ plane features to be selected; N_e^*/N_P^* number of edge / plane features already selected; p_ε/p_S edge / plane feature points.
Input: $p^N, p \in p^N$
Output: p_ε, p_S

```
1   for every Rang ID $i = 0$ to $n$ do
2     $\delta, \varepsilon \leftarrow$ PCA calculation features $(p_i)$
3     $\chi_e, \chi_p \leftarrow 1 - \gamma_t R_i, t = (edge, plane)$
4     $N_e, N_p \leftarrow \omega_t R_i$
5     if $\varepsilon > \chi_e$ then
6       if $N_e^* < N_e$ then
7         $p_\varepsilon \leftarrow p_i$
8         $N_e^* = N_e^* + 1$
9       end
10    else if $\delta > \chi_p$ then
11      if $N_p^* < N_p$ then
12        $p_S \leftarrow p_i$
13        $N_p^* = N_p^* + 1$
14    end
15    end
16  end
17  OUTPUT: $p_\varepsilon, p_S$
```

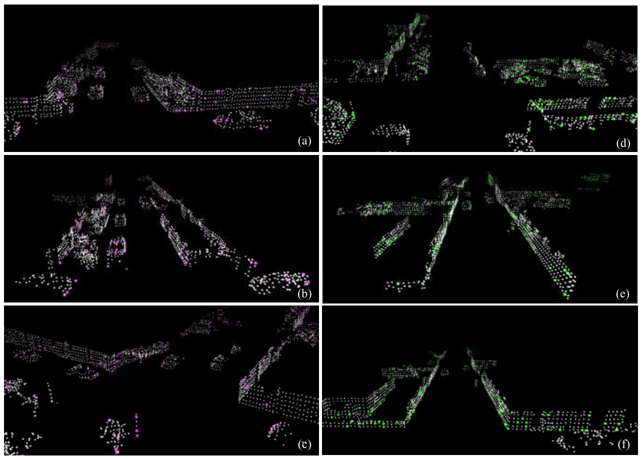


FIGURE 4. Extraction of edge and plane features from non-ground point clouds. (a)-(c) Edge feature points, (d)-(f) plane feature points.

E. UPDATE MODEL

Pose estimation is to align the current edge feature points and plane feature points in the world coordinate of global map. In order to speed up the search of corresponding points, edge and plane feature points are stored in KD-tree. The world

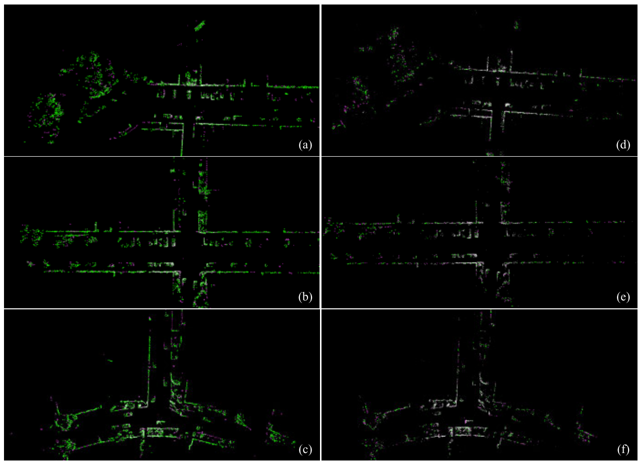


FIGURE 5. Comparison on different feature point selections. (a)-(c) Fixed selection of feature points, (d)-(f) adaptive selection of feature points.

coordinate system is marked as w and the LiDAR coordinate system is marked as $\mathcal{L}$. The state transition relationship of a vehicle can be expressed as:

$$\tilde{\mathcal{P}}_k^w = T_{\mathcal{L}}^w \otimes \mathcal{P}_k^{\mathcal{L}}. \tag{13}$$

By the Lie algebra, the state transformation of the six degrees of freedom is expressed as follows:

$$\xi_k^{w\wedge} = \begin{bmatrix} \boldsymbol{\phi}_\times & \boldsymbol{\rho} \\ \mathbf{0}^T & 0 \end{bmatrix} \in \mathfrak{se}(3), \quad \xi_k^{w\wedge} = [\boldsymbol{\rho},\ \boldsymbol{\phi}]^T \in \mathbb{R}^6, \tag{14}$$

where ρ and ϕ respectively represent the corresponding translation and rotation in the tangent vector,

$$\boldsymbol{\phi}_\times = \begin{bmatrix} 0 & -\phi_3 & \phi_2 \\ \phi_3 & 0 & -\phi_1 \\ -\phi_2 & \phi_1 & 0 \end{bmatrix} \in \mathbb{R}^{3\times 3}. \tag{15}$$

Lie algebras and Lie groups are transformed by exponential mapping and logarithmic mapping [24]. The transformation of Lie group $T_{\mathcal{L}}^w = [\boldsymbol{R},\ \boldsymbol{t}] \in SE(3)$ contains a rotation matrix $\boldsymbol{R} \in SO(3)$ and a translation vector $\boldsymbol{t} \in \mathbb{R}^3$. The mapping relationship between the rotation matrix and the translation vector is:

$$T_{\mathcal{L}}^w = \exp\left(\xi_k^w\right) = \begin{bmatrix} \exp\left(\boldsymbol{\phi}_\times\right) & \mathcal{J}\boldsymbol{\rho} \\ \mathbf{0}^T & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & \boldsymbol{t} \\ \mathbf{0}^T & 1 \end{bmatrix}, \tag{16}$$

where $\mathcal{J}$ is the Jacobian matrix [26], which is defined as:

$$\mathcal{J} = \frac{\sin\theta}{\theta}\boldsymbol{I} + \left(1 - \frac{\sin\theta}{\theta}\right)\boldsymbol{a}\boldsymbol{a}^T + \frac{1-\cos\theta}{\theta}\boldsymbol{a}_\times, \tag{17}$$

θ is the module length of ξ_k^w, and $\boldsymbol{a}$ is a direction vector of ξ_k^w with a length of 1, $||\boldsymbol{a}|| = 1$.

$$\xi_k^w = \log\left(T_{\mathcal{L}}^w\right) = \begin{bmatrix} \mathcal{J}^{-1}\boldsymbol{t} \\ \log\left(\mathbf{R}\right) \end{bmatrix}. \tag{18}$$

The status update model of the vehicle is as follows:

$$\tilde{\xi_k^w} = \log\left(\exp\left(\xi_k^w\right) \cdot \exp\left(\delta\xi_k^w\right)\right), \tag{19}$$

$\delta\xi_k^w$ is the increment of pose, which is obtained through continuous iterative optimization.

F. POSE ESTIMATION

The point cloud is transformed into the same coordinate system to estimate odometer pose. The edge features $\mathcal{P}_{\varepsilon k}^{\ell} \epsilon \mathbb{P}$ are transformed to the world coordinate system $\tilde{\mathcal{P}}_{\varepsilon k}^w$. Two edge points $(\mathcal{P}_a^w,\ \mathcal{P}_b^w)$ need to be found in the world coordinate system. These two points are connected into a straight line to construct point-to-line residuals. First, five nearby edge points are searched in the world coordinate system. Then, the local center points of the five points are calculated according to equation (7). Finally, the point in front of the center is selected using $\mathcal{P}_a^w$ and the point behind is selected $\mathcal{P}_b^w$ using. The distance residual equation from point to line is constructed as follows:

$$\mathcal{R}_\varepsilon = \frac{\left(\tilde{\mathcal{P}}_{\varepsilon k}^w - \mathcal{P}_{\varepsilon b}^w\right) \times \left(\tilde{\mathcal{P}}_{\varepsilon k}^w - \mathcal{P}_{\varepsilon a}^w\right)}{\left|\mathcal{P}_{\varepsilon a}^w - \mathcal{P}_{\varepsilon b}^w\right|}. \tag{20}$$

Similarly, the plane feature point $\mathcal{P}_{\mathcal{S}k}^{\ell} \epsilon \mathbb{P}$ are transformed to the world coordinate system $\tilde{\mathcal{P}}_{\mathcal{S}k}^w$. Three nearby plane points $(\mathcal{P}_{\mathcal{S}_t}^w, \mathcal{P}_{\mathcal{S}_u}^w, \mathcal{P}_{\mathcal{S}_v}^w)$ are searched in the world coordinate system to form a plane. The residual equation of the distance from the point to the plane is as follows:

$$\mathcal{R}_\mathcal{S} = \left(\tilde{\mathcal{P}}_{\mathcal{S}k}^w - \mathcal{P}_{\mathcal{S}_t}^w\right) \cdot \frac{\left(\mathcal{P}_{\mathcal{S}_u}^w - \mathcal{P}_{\mathcal{S}_t}^w\right) \times \left(\mathcal{P}_{\mathcal{S}_v}^w - \mathcal{P}_{\mathcal{S}_t}^w\right)}{\left|\left(\mathcal{P}_{\mathcal{S}_u}^w - \mathcal{P}_{\mathcal{S}_t}^w\right) \times \left(\mathcal{P}_{\mathcal{S}_v}^w - \mathcal{P}_{\mathcal{S}_t}^w\right)\right|}. \tag{21}$$

We combine line residuals and plane residuals to obtain the loss function for pose optimization as follows:

$$\boldsymbol{F}(T) = \min_{T \in SE(3)} f\left(\sum \mathcal{R}_\varepsilon + \sum \mathcal{R}_\mathcal{S}\right), \tag{22}$$

The nonlinear optimization problem is solved by Gauss Newton method. A first-order Taylor expansion is carried out to solve the increment as follows: $f\ (x + \Delta x) \approx f(x) + \mathcal{J}(x)\Delta x$. The incremental equation is rewritten as follows:

$$\mathcal{H}\Delta x = g, \tag{23}$$

here $\mathcal{J}$ is Jacobian matrix, $\mathcal{H}$ is a Hessian matrix defined as $\mathcal{H} = \mathcal{J}(x)\mathcal{J}^T(x)$, Δx is the incremental amount, and $g = -\mathcal{J}(x)f(x)$. The nonlinear problem is transformed into iterative solution increment Δx. The Jacobian matrix $\mathcal{J}_\varepsilon$ of edge residual can be calculated by:

$$\begin{aligned} \mathcal{J}_\varepsilon &= \frac{\partial \mathcal{R}_\varepsilon}{\partial T} = \frac{\partial \mathcal{R}_\varepsilon}{\partial \tilde{\mathcal{P}}_{\varepsilon k}^w}\frac{\partial \tilde{\mathcal{P}}_{\varepsilon k}^w}{\partial T} \\ &= -\frac{\left(\left(\tilde{\mathcal{P}}_{\varepsilon k}^w - \mathcal{P}_{\varepsilon b}^w\right) \times \left(\tilde{\mathcal{P}}_{\varepsilon k}^w - \mathcal{P}_{\varepsilon a}^w\right)\right)^T}{\left|\left(\tilde{\mathcal{P}}_{\varepsilon k}^w - \mathcal{P}_{\varepsilon b}^w\right) \times \left(\tilde{\mathcal{P}}_{\varepsilon k}^w - \mathcal{P}_{\varepsilon a}^w\right)\right|} \\ &\quad \cdot \frac{\left(\mathcal{P}_{\varepsilon a}^w - \mathcal{P}_{\varepsilon b}^w\right)_\times}{\left|\mathcal{P}_{\varepsilon a}^w - \mathcal{P}_{\varepsilon b}^w\right|} \cdot \mathcal{J}_C, \end{aligned} \tag{24}$$

the Jacobian $\mathcal{J}_C$ can be estimated by applying left perturbation model with $\delta\xi \epsilon\ \mathfrak{se}(3)$

$$\begin{aligned} \mathcal{J}_{\varepsilon C} &= \frac{\partial \tilde{\mathcal{P}}_{\varepsilon k}^w}{\partial \delta\xi} = \lim_{\delta\xi \to 0} \frac{\exp\left(\delta\xi_\times\right)\tilde{\mathcal{P}}_{\varepsilon k}^w - \tilde{\mathcal{P}}_{\varepsilon k}^w}{\delta\xi} \\ &= \begin{bmatrix} \boldsymbol{I} & -\left[\tilde{\mathcal{P}}_{\varepsilon k}^w\right]_\times \\ \mathbf{0}^T & \mathbf{0}^T \end{bmatrix}. \end{aligned} \tag{25}$$

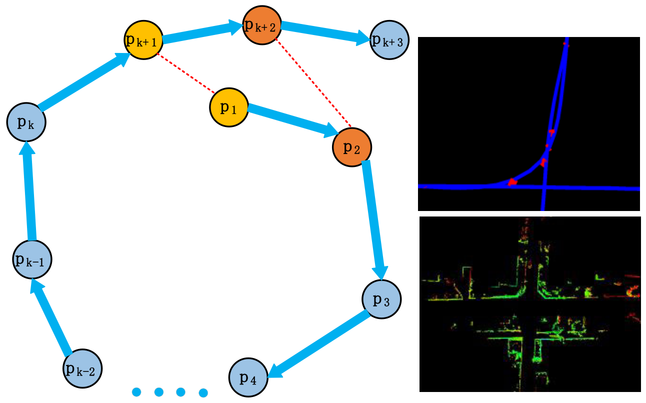


FIGURE 6. **Graph-based optimization of loop detection. The left figure is the diagram of the loop detection, the right top figure is the trajectory of the loop detected by the algorithm and the right bottom is the point cloud mapping.**

Similarly, the plane Jacobian matrix can be obtained

$$\mathcal{J}_{\mathcal{S}} = \frac{\partial \mathcal{R}_{\mathcal{S}}}{\partial T} = \frac{\left(\mathcal{p}^w_{\mathcal{S}_u} - \mathcal{p}^w_{\mathcal{S}_t}\right) \times \left(\mathcal{p}^w_{\mathcal{S}_v} - \mathcal{p}^w_{\mathcal{S}_t}\right)^T}{\left|\left(\mathcal{p}^w_{\mathcal{S}_u} - \mathcal{p}^w_{\mathcal{S}_t}\right) \times \left(\mathcal{p}^w_{\mathcal{S}_v} - \mathcal{p}^w_{\mathcal{S}_t}\right)\right|} \cdot \mathcal{J}_{\mathcal{S}C}. \qquad (26)$$

The incremental equation is solved by the Jacobian matrix, of which increments is optimized iteratively until the equation converges.

G. BACK-END OPTIMIZATION

The long-term odometer continuously produces cumulative errors, resulting in poor global mapping. The cumulative error can be eliminated by loop detection and then global optimization of the map. In order to accelerate the map optimization, we adopt the approach based on key frames during loop detection and global optimization. When the attitude change between two frames exceeds a certain threshold, the current frame is selected as the key frame. A frame similar to the current frame is omitted in the historical key frame. The relative poses of two similar frames are added to the graph optimization as constrained edges. We use a two-step loop detection method. Firstly, the fast and efficient loop detection approach Scan Context [27] is used to find the closed-loop candidate frame from the historical key frame. Scan Context introduces a ''rotation invariance'' descriptor to quickly detect loops occurring in different directions. Then, ICP is used to match the current frame with the candidate frame to get the score between the two frames. As shown in Figure 6, a loop occurs in two frames if the score is less than a preset threshold. The relative position between the two frames of the loop is added to the graph optimization system GTSAM [28] as a constrained edge. This optimization system can effectively optimize mapping to eliminate cumulative errors. Historical position and global mapping are updated accordingly.

V. EXPERIMENTAL EVALUATION

A. TESTING FRONT-END ODOMETRY IN KITTI

We first validate the accuracy and effectiveness of the front-end odometer in the proposed system on the KITTI odometry benchmark [15]. The data acquisition platform for the KITTI dataset is fitted with two grey-scale cameras, two color cameras, a Velodyne HDL-64E LIDAR, and a GPS navigation system. In the testing, only data from the LIDAR were used. The dataset was collected from large complex scenes including urban, rural and highway. Sequences 00-10, which provided ground truth values, were chosen to evaluate the algorithm. There are 23,201 frames and 22 km of track length in 11 sequences. The accuracy of the odometer was tested using the odometer metric provided by KITTI. The translation and rotation errors were calculated for different lengths, specifically, every 100 m up to 800 m. The average of the errors ARE and ATE defined by the KITTI odometry benchmark are indicators of the average rotational error and average translation error, respectively. The smaller value indicates better accuracy of the algorithm.

The proposed algorithm is compared with typical state-of-the-art odometers ALOAM, FLOAM [20] and SUMA [29]. The results of ALOAM are from the paper [12] and the results of SUMA are from its original paper. In Table 1, the average translation error of our method on 11 sequences is 59% smaller than SUMA, 71% smaller than ALOAM and 19% smaller than FLOAM. The average rotation error is 42% smaller than SUMA, 69% smaller than ALOAM and 7.1% smaller than FLOAM. The comparison shows that ours is more accurate than the above methods. The localization error of our algorithm is smaller, especially in large urban environments such as the 00 sequence (The trajectory length of 00 sequence is 3741m).

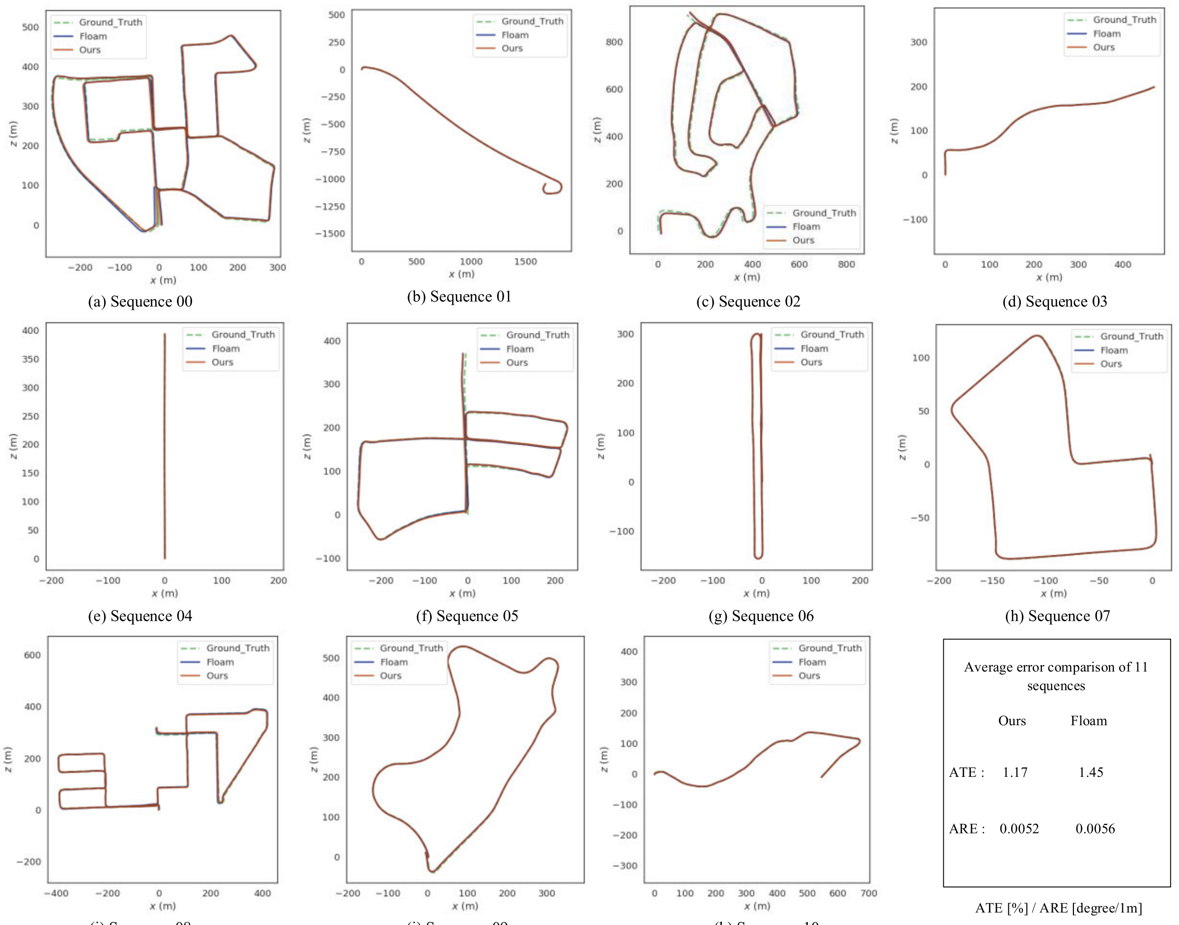
The proposed odometry method (without the use of loop detection) is compared with the state-of-the-art algorithm FLOAM on the trajectory. Figure 7 shows a comparison of three trajectories among our algorithm, FLOAM and ground truth on 11 sequences. The solid red line is the track of our algorithm, the solid blue line is the FLOAM odometer track, and the dotted green line is the ground truth from GPS/INS. As the proposed algorithm removes some redundant points. The extracted feature points are more uniformly distributed in six degrees of freedom. Therefore, our algorithm is closer to the ground truth than FLOAM on most sequences. All trajectories essentially coincide with the ground truth. Figure 8 shows the accuracy of the proposed odometer estimate, we compare the absolute pose errors (APE) estimated by the odometer between FLOAM and the proposed algorithm. The APE of the proposed algorithm is smaller than FLOAM, our algorithm is more accurate. The error has a smaller range of fluctuations, our system is more stable. The proposed algorithm is also smaller than FLOAM in other evaluation metrics, including root mean square error (RMSE) and standard deviation (STD). Experiments prove that the proposed algorithm can accurately locate in large-scale urban environment.

B. SLAM SYSTEM ROBUSTNESS EXPERIMENTS

We have chosen three representative urban scenario sequences 00, 05 and 07 from the KITTI odometry benchmark to analyze. Figure 9 shows the results of the mapping

TABLE 1. LiDAR Odometry evaluation comparison on KITTI dataset. All errors are represented as ATE [%] / ARE [degree/1m] (the smaller the better). Loop detection is not used in all methods.

Sequence	Environment	Path Length (m)	SuMa		ALOAM		FLOAM		Ours	
			ATE	ARE	ATE	ARE	ATE	ARE	ATE	ARE
00	Urban	3714	2.10	0.0090	4.82	0.0190	1.36	0.0059	**0.78**	**0.0042**
01	Highway	4268	4.00	0.0120	3.90	0.0108	2.63	0.0056	**2.34**	**0.0052**
02	Urban+Country	5075	2.30	0.0080	5.50	0.0240	1.84	0.0062	**1.36**	**0.0052**
03	Country	563	**1.40**	0.0070	5.50	0.0241	1.51	0.0073	1.42	**0.007**
04	Country	397	11.9	0.0110	1.51	0.0115	1.19	0.0039	**1.08**	**0.0037**
05	Urban	2223	1.50	0.0080	4.80	0.0192	0.88	**0.0036**	**0.80**	0.0045
06	Urban	1239	1.00	0.0060	3.10	0.0112	0.65	0.0037	**0.56**	**0.0034**
07	Urban	695	1.80	0.0120	2.94	0.0179	1.16	0.0069	**0.64**	**0.0063**
08	Urban+Country	3225	2.50	0.0100	4.77	0.0208	**1.44**	0.0057	1.45	**0.0055**
09	Urban+Country	1717	1.90	0.0080	5.73	0.0186	1.50	0.0062	**1.29**	**0.0056**
10	Urban+Country	919	1.80	0.0100	3.48	0.0139	1.85	0.0068	**1.20**	**0.0061**
Average			2.92	0.0091	4.18	0.0173	1.45	0.0056	**1.17**	**0.0052**



(a) Sequence 00 (b) Sequence 01 (c) Sequence 02 (d) Sequence 03

(e) Sequence 04 (f) Sequence 05 (g) Sequence 06 (h) Sequence 07

(i) Sequence 08 (j) Sequence 09 (k) Sequence 10

Average error comparison of 11 sequences

	Ours	Floam
ATE :	1.17	1.45
ARE :	0.0052	0.0056

ATE [%] / ARE [degree/1m]

FIGURE 7. The proposed front-end odometer, FLOAM and KITTI ground truth track alignment. (a)-(k) correspond to the sequence 00-10 of KITTI, respectively.

created by the SLAM system in a large urban environment with KITTI 00 sequences, and the errors in the position and direction of the mapping. The environment 00 sequence is characterized by a large urban area and a complex environment, containing most typical urban scenes with moving objects such as moving vehicles, cyclists and pedestrians,

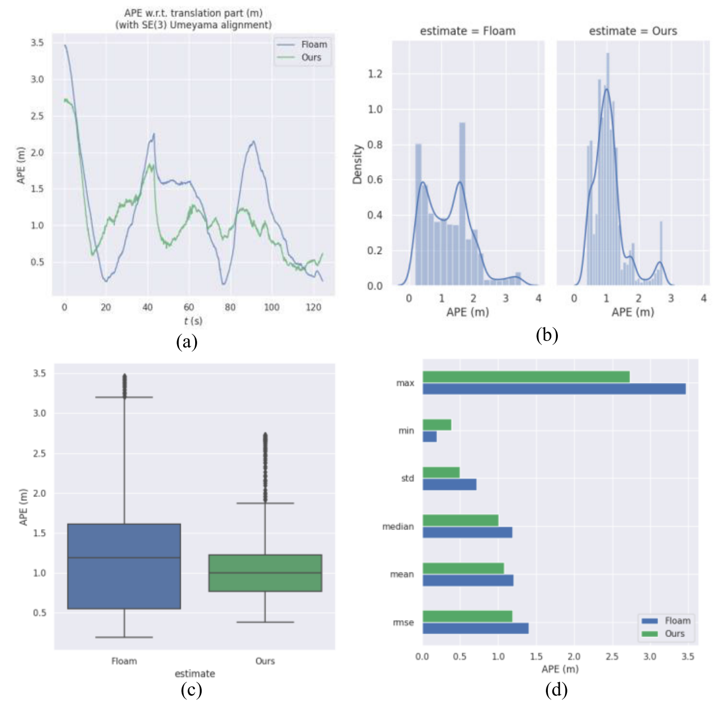


FIGURE 8. Absolute pose error (APE) of the odometry estimation comparison between FLOAM and ours on KITTI 10 sequence.

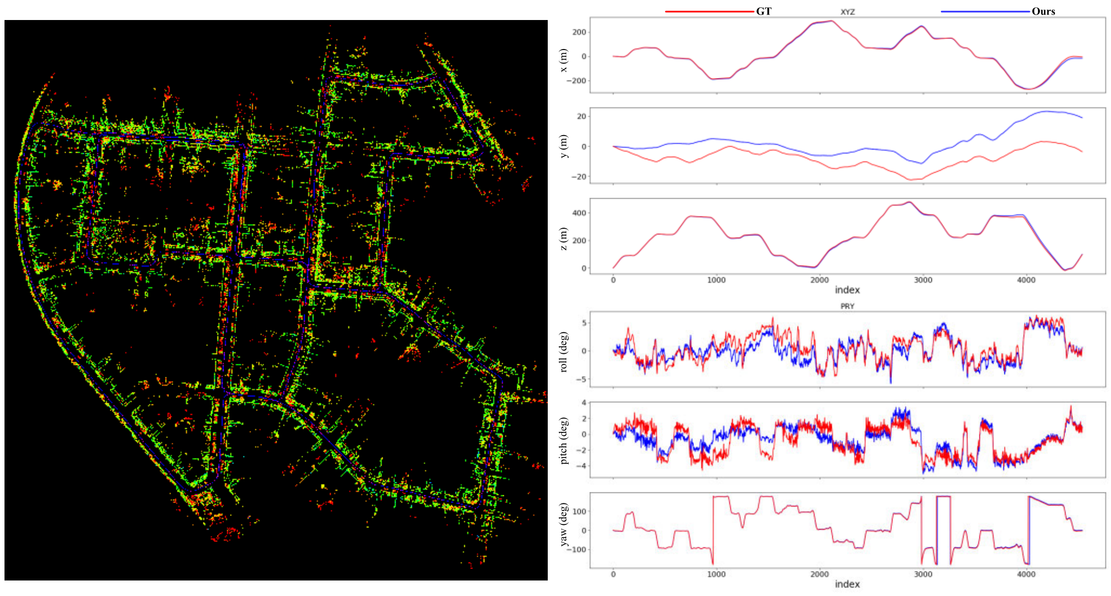


FIGURE 9. The left figure shows the point cloud mapping created by the proposed system on the KITTI sequence 00, the right ones show the errors of the positions and orientations compared with the ground truth. (The xyz coordinate system has been transformed into the zxy coordinate system).

all of which have impacts on positioning accuracy. The proposed algorithm is able to cope with the above scenarios with moving objects. The localization error of the algorithm is small in the x, y and yaw directions. As shown in Figure 9, the 2D pose components in x, y, and yaw are approaching to the ground truth, while some errors exist in altitude, roll, and pitch directions. The errors can be attributed to the difficulty of capturing slope and transient directional changes by LiDAR. The localization accuracy is less affected in flat urban environments. Therefore, the proposed SLAM system achieves accurate localization in large-scale urban environment.

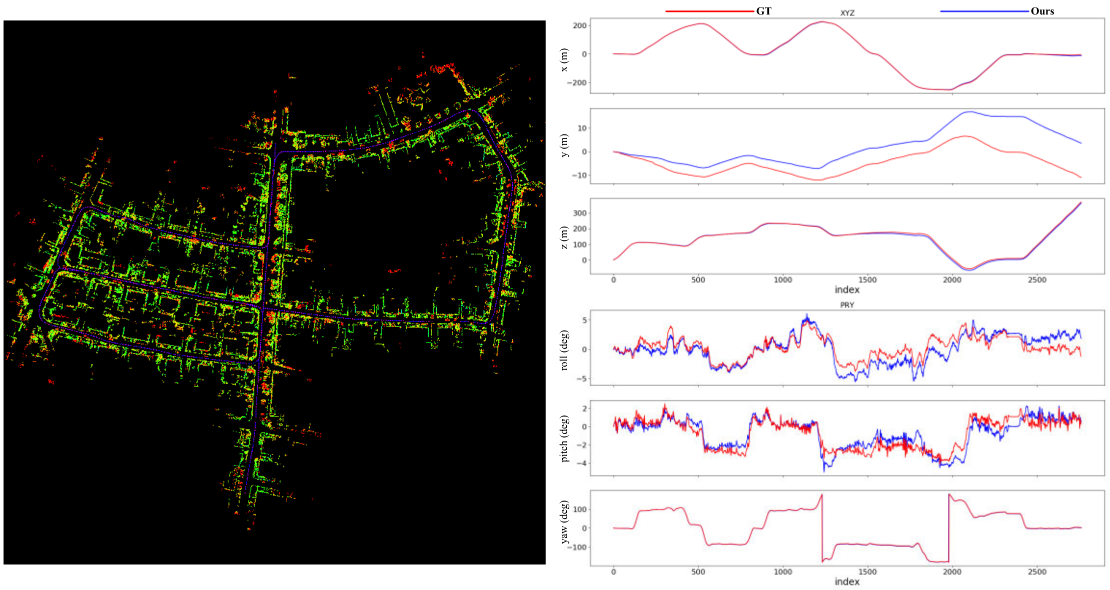


FIGURE 10. The left figure shows the point cloud mapping created by the proposed system on the KITTI sequence 05, the right ones show the errors of the positions and orientations compared with the ground truth. (The xyz coordinate system has been transformed into the zxy coordinate system).

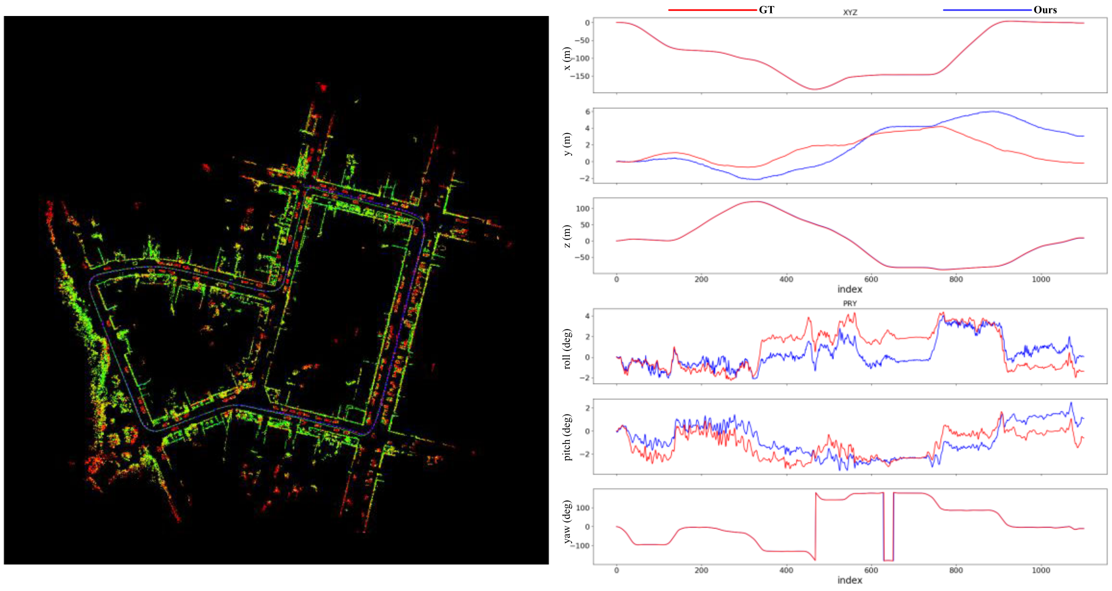


FIGURE 11. The left figure shows the point cloud mapping created by the proposed system on the KITTI sequence 07, the right ones show the errors of the positions and orientations compared with the ground truth. (The xyz coordinate system has been transformed into the zxy coordinate system).

The trajectory of the vehicle is curved in the KITTI 05 sequence with a length of 2,223 m. The vehicle passing the same crossroads from different directions leads to the difficulty of loop detection. Scan Context can effectively detect loops in the above scenario due to the induction of rotation invariants descriptor. As shown in Figure 10, the proposed system can build 3D point cloud map in KITTI 05 scenario which is basically consistent with the real environment. The localization accuracies in x, y and yaw directions approach the ground truth. As shown in Figure 11, sequence 07 is another urban scenario different from sequence 05. The vehicle circles the urban and comes back to its origin in sequence 07. The proposed system locates accurately and builds low-drift 3D point cloud maps in this scenario. The testing results show that the system can accurately locate and mapping in three different urban scenarios of KITTI.

To verify feasibility of the feature extraction algorithm on LiDAR with different resolutions. The proposed algorithm

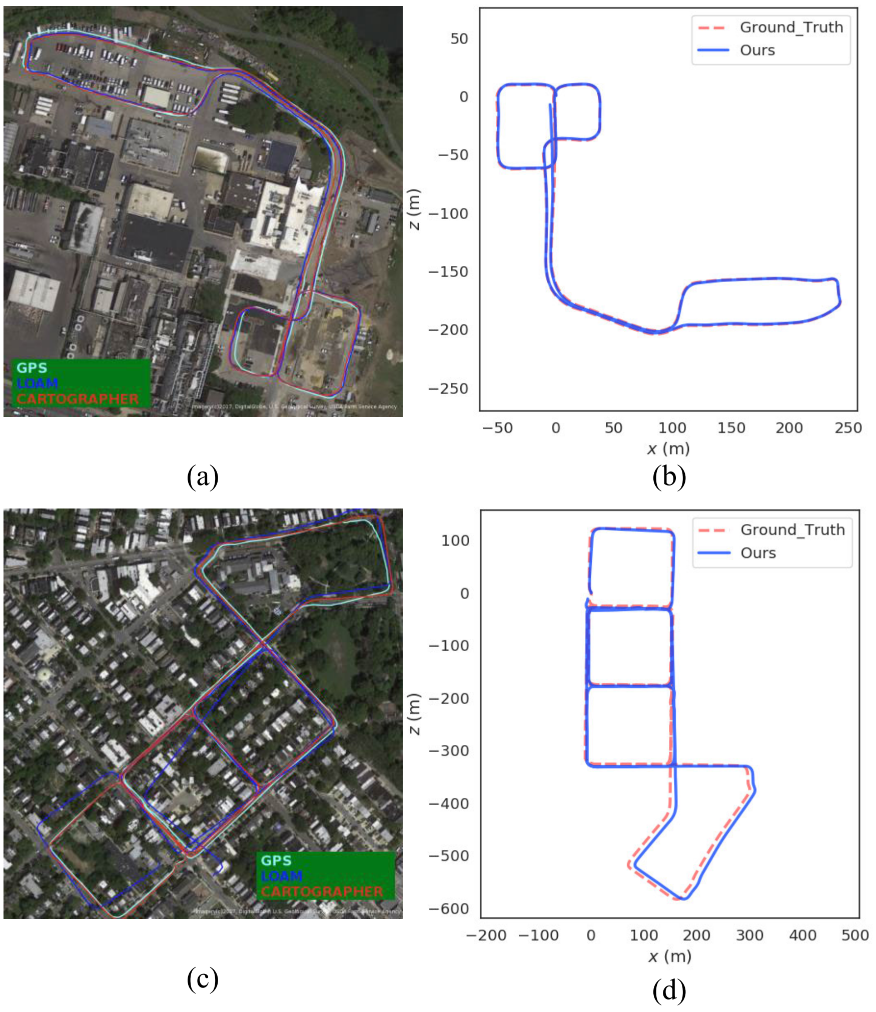


FIGURE 12. Trajectory deviations of GPS, Cartographer and LOAM overlaid on satellite images of (a) Day 1 and (c) Day 2 from the MVSECD dataset. Comparisons of our algorithm's trajectory with the ground truth trajectories on the MVSECD, (b) Day 1 and (d) Day2.

was tested on MVSECD [16] datasets MVSECD, in which the horizontal and vertical resolutions of LiDAR are different from KITTI's. As shown in Figure 12, The solid blue line is the trajectory of our algorithm, and the red line is the ground truth. Our system achieves excellent positioning accuracy in different urban scenarios. The trajectories obtained by the proposed system are approaching to the ground truth in Day 1 and 2, which verifies the effectiveness of the feature point extraction method using depth information without dependence on LiDAR resolution and the robustness of the proposed SLAM system applied in complex urban environments.

C. RUNTIME ANALYSIS

The real-time performance of the proposed SLAM system was tested on KITTI sequence 00. The running time of our system is compared with that of ALOAM and FLOAM. We conducted the experiments on a PC platform with Intel Core i7-10700 2.90GHz CPU, the testing environment is based on ROS Melodic and Ubuntu 18.04 LTS. In the front-end, the running time of ground segmentation, feature extraction and range attitude estimation were tested respectively. In the back-end, loop detection and pose graph optimization were combined as one module to test the time. As he results shown in Table 2, the average runtimes of running ground segmentation module, feature extraction module, and odometer position estimation module are 26, 29, and 21 ms, respectively. The whole front-end costs an average of 76 ms. The proposed feature extraction algorithm using adaptive selection manner extracts fewer points compared with ALOAM and FLOAM. Therefore, the pose estimation in our system costs less time. The running time of the front-end odometer in our system is 23 ms faster than that of ALOAM. The front-ends in the proposed system and FLOAM cost almost equal time. As the number of point clouds increases during the mapping process, the mapping time of FLOAM and ALOAM are increasing. However, our system has fewer

TABLE 2. Runtime comparison and analysis.

Methods	Front-end			Back-end
	Ground segmentation	Feature extraction	Position estimation	Loop closure
ALOAM	-	32 ms	67 ms	-
FLOAM	-	27 ms	45 ms	-
Ours	26 ms	29 ms	21 ms	290 ms

feature points, so the real-time mapping is better.[1] For the back-end, loop occurs after the vehicle running for some time. The map is optimized only when the loops occur. We use a lower frequency of 2 Hz to detect loop. The whole back-end costs an average of 290 ms.

The back-end costs more time compared to the front-end due to the need of optimizing and updating the global map and poses. It is necessary to be noted that the optimization and update of global map performs only when the loop occurs rather than every frame. Therefore, its effect on the real-time performance of the system can be neglected. The testing results show that the proposed SLAM system has achieved excellent real-time performance in all the above testing. The system can be applied to autonomous vehicles with resource constraint devices.

VI. CONCLUSION

A lightweight real-time 3D LiDAR SLAM system is presented to solve the localization and mapping problems of autonomous vehicles in large and complex urban environments. The proposed system consists of ground segmentation, depth encoding, features extraction based on modified PCA and loop detection. The disordered non-ground point clouds are encoded according to depth information. The encoded point clouds maintain three-dimensional information without being projected onto a 2D plane. This coding approach can be applied to LiDAR with different resolutions. The adaptive selection approach of neighbor points in modified PCA improves the speed of feature extraction. Uniformly distributed points can be extracted in six degrees of freedom to increase the localization accuracy of the odometer by selecting different number of feature points according to the distance. The average translation error of the odometer is only 1.17% and the average rotation error is only 0.052 (degree/1m) on KITTI dataset. Due to sparse points extracted by the adaptive feature extraction approach, the pose estimation in the odometer costs only 21 ms. The whole front-end costs 76 ms, achieving real-time performance. Scan Context and ICP are employed for loop detection in order to eliminate mapping accumulation errors. A graph-based optimization method is used to optimize the global mapping. To demonstrate the robustness of the proposed system in different urban scenarios, the performance of the system was evaluated on KITTI and MVSECD datasets. The localization accuracy of the system can approach the ground truth in different scenarios of both above datasets.

[1]https://youtu.be/jnL7UMFvzbU

REFERENCES

[1] M. Elhousni and X. Huang, "A survey on 3D LiDAR localization for autonomous vehicles," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Oct. 2020, pp. 1879–1884.

[2] M. Kovacova, J. Oláh, J. Popp, and E. Nica, "The algorithmic governance of autonomous driving behaviors: Multi-sensor data fusion, spatial computing technologies, and movement tracking tools," *Contemp. Readings Law Social Justice*, vol. 14, no. 2, pp. 27–45, 2022, doi: 10.22381/CRLSJ14220222.

[3] M. Nagy and G. Lăzăroiu, "Computer vision algorithms, remote sensing data fusion techniques, and mapping and navigation tools in the industry 4.0-based Slovak automotive sector," *Mathematics*, vol. 10, no. 19, p. 3543, Sep. 2022, doi: 10.3390/MATH10193543.

[4] K. Valaskova, J. Horak, and G. Lăzăroiu, "Socially responsible technologies in autonomous mobility systems: Self-driving car control algorithms, virtual data modeling tools, and cognitive wireless sensor networks," *Contemp. Readings Law Social Justice*, vol. 14, no. 2, pp. 172–188, 2022, doi: 10.22381/CRLSJ142202210.

[5] 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.

[6] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source slam system for monocular, stereo, and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017.

[7] T. Qin, P. Li, and S. Shen, "VINS-Mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 1004–1020, Aug. 2018.

[8] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real time," in *Proc. Robot., Sci. Syst. Conf. (RSS)*, vol. 2, no. 9, Berkeley, CA, USA, 2014.

[9] 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.

[10] K. Koide, J. Miura, and E. Menegatti, "A portable three-dimensional LiDAR-based system for long-term and wide-area people behavior measurement," *Int. J. Adv. Robot. Syst.*, vol. 16, no. 2, pp. 1–16, Apr. 2019.

[11] S. Zhao, Z. Fang, H. Li, and S. Scherer, "A robust laser-inertial odometry and mapping method for large-scale highway environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 1285–1292.

[12] P. Zhou, X. Guo, X. Pei, and C. Chen, "T-LOAM: Truncated least squares LiDAR-only odometry and mapping in real time," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2022, Art. no. 5701013.

[13] P. J. Besl and D. N. McKay, "A method for registration of 3-D shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 239–256, Feb. 1992.

[14] 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.

[15] 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.

[16] A. Z. Zhu, D. Thakur, T. Ozaslan, B. Pfrommer, V. Kumar, and K. Daniilidis, "The multivehicle stereo event camera dataset: An event camera dataset for 3D perception," *IEEE Robot. Autom. Lett.*, vol. 3, no. 3, pp. 2032–2039, Jul. 2018.

[17] A. Segal, D. Haehnel, and S. Thrun, "Generalized-ICP," in *Robotics: Science and Systems*, vol. 2, no. 4. Seattle, WA, USA: MIT Press, 2009, p. 435.

[18] 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.

[19] 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. 5135–5142.

[20] H. Wang, C. Wang, C.-L. Chen, and L. Xie, "F-LOAM : Fast LiDAR odometry and mapping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2021, pp. 4390–4396.

[21] 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.

[22] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with application to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[23] M. Himmelsbach, F. V. Hundelshausen, and H.-J. Wuensche, "Fast segmentation of 3D point clouds for ground vehicles," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2010, pp. 560–565.

[24] 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. Autom. (ICRA)*, May 2017, pp. 5067–5073.

[25] 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, Jul. 2016.

[26] A. Kirillov Jr., *An Introduction to Lie Groups and Lie Algebras*, vol. 113. Cambridge, U.K.: Cambridge Univ. Press, 2008.

[27] 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.

[28] F. Dellaert, "Factor graphs and GTSAM: A hands-on introduction," Georgia Inst. Technol., Atlanta, GA, USA, Tech. Rep., GT-RIM-CP&R-2012-002, Sep. 2012.

[29] J. Behley and C. Stachniss, "Efficient surfel-based SLAM using 3D laser range data in urban environments," in *Robotics: Science and Systems*. Pittsburgh, PA, USA: Carnegie Mellon Univ., 2018.

GUANGMAN LU received the bachelor's degree in communication engineering from Guangxi University for Nationalities. He is currently pursuing the master's degree with the School of Electronics and Communication Engineering, Guangzhou University. His current research interests include the simultaneous localization and mapping based on 3D LiDAR.

HONG YANG was born in Hunan, China. He received the B.S. degree in computer engineering from the Xi'an University of Electronic Science and Technology, Xi'an, China, in 1988, and the M.S. degree in circuits and systems and the Ph.D. degree in control theory and control engineering from the South China University of Technology, Guangzhou, China, in 2004 and 2010, respectively. Since 2004, he has been an Associate Professor with the College of Mechanical and Electrical Engineering, Guangzhou University, Guangzhou. His current research interests include PMSM control and deep learning.

JIE LI received the bachelor's degree in electronic information engineering from Suzhou University, in 2020. He is currently pursuing the master's degree with the School of Electronics and Communication Engineering, Guangzhou University, Guangzhou, China. His current research interests include the simultaneous localization and mapping based on vision.

ZHENFEI KUANG received the bachelor's degree in measurement and control technology and instrumentation from the Nanchang Institute of Technology, in 2020. He is currently pursuing the master's degree with the School of Electronics and Communication Engineering, Guangzhou University. His current research interests include visual inertial simultaneous localization and mapping (VI-SLAM).

RU YANG was born in Hunan, China. She received the M.S. degree in electronic engineering from the Guangdong University of Technology, Guangzhou, China, in 1999, and the Ph.D. degree in power electronics from the South China University of Technology, Guangzhou, in 2004. Since 1999, she has been with Guangzhou University, Guangzhou, where she is currently a Professor with the School of Physics and Electronic Engineering. She has authored more than 30 published technical papers. Her current research interests include time and frequency-domain analysis of power electronics, chaotic switching techniques, and electromagnetic interference.