FEVO-LOAM: Feature Extraction and Vertical Optimized Lidar Odometry and Mapping

Zelin Wang, Limin Yang, Feng Gao, and Liangyu Wang

***Abstract*—Simultaneous Localization and Mapping (SLAM) is a significant research topic in robotics since it is one of the key technologies for robot automation. Although lidar-based SLAM methods have achieved promising performance, traditional lidar SLAM methods still produce large vertical errors. To address this issue, we propose a feature extraction and vertical optimized lidar odometry and mapping approach. Firstly, we optimize the feature extraction. Specifically, we propose a more accurate ground segmentation approach and a new curvature definition, which is used to extract more discriminative features. Additionally, we propose a lidar mapping approach, which adds new vertical residuals and pitch residuals to the objective function. Then a two-step Levenberg-Marquardt method is used to solve the pose transformation. Finally, we evaluate the proposed method in public datasets and real environments. Experiments show that compared with other state-of-the-art methods, our method achieves better accuracy and reduces the vertical error with a similar computational expense.**

***Index Terms*—SLAM, mapping, robot automation.**

在目标函数中添加了新的垂直残差和俯仰残差

I. Introduction

SIMULTANEOUS Localization and Mapping (SLAM) is one of the popular research topics in robotics in recent years. It mainly solves the problem of localization, navigation, and mapping when mobile robots are working in unknown environments. The existing literature uses a variety of sensors to solve the SLAM problems, such as vision-based [1], magnetic-field based [2], sonar-based [3] and lidar-based [4]. Compared with others, lidar SLAM is less affected by the environment, such as illumination and viewpoint changes. In addition, it provides high-precision point clouds and is more likely to achieve higher localization and mapping accuracy [5]. Therefore, lidar SLAM has a wide range of robotic applications, such as unmanned aerial vehicles [6], autonomous driving [7], and mobile robots [8].

Generally, the key of lidar SLAM is to find the pose transformation between the current scan and the reference point cloud, which is also known as scan matching. The most classic solution is the Iterative Closed Point (ICP) algorithm [9], It targets the minimum distance between two point clouds and iterates until it converges or satisfies the stopping criterion. However, the amount of point cloud data of lidar is huge, and ICP may suffer from prohibitive computational cost, thus it is difficult to meet the real-time requirements. In addition, ICP also requires good initialization for better convergence.

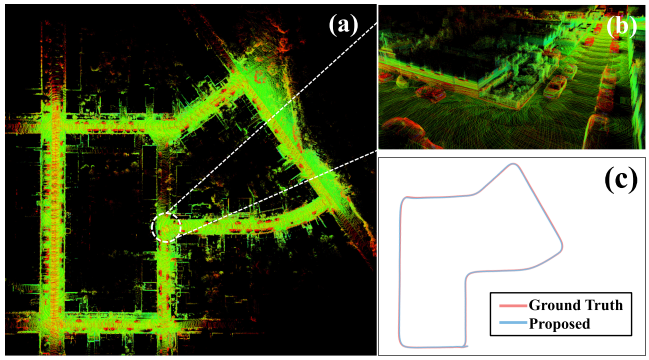


Fig. 1. Experimental results of the proposed method on KITTI dataset sequence 07. (a) Mapping result. (b) Details of the result. (c) Comparison with ground truth.

由于边缘点和平面点的残差约束对垂直位姿变换不敏感，这些方法在垂直方向上会产生较大的误差

Other methods are the feature-based matching methods. It improves computational efficiency by extracting representative feature points in the point cloud. Lidar Odometry And Mapping (LOAM) [10] is a typical example. It divides the feature points into edge features and plane features, and it calculates the pose transformation by minimizing point-to-plane and point-to-edge distance. However, these methods will produce large errors in the vertical direction, since the residual constraints of edge and plane points are not sensitive to vertical pose transformation.

To address the above limitations, we propose an efficient and high-accuracy lidar SLAM framework, which targets to reduce the vertical error of lidar SLAM. Fig. 1 shows the mapping results of our method. The main contributions of our work are:

1) We propose a new ground segmentation approach. It takes the distance from the point to the lidar and the normal direction of each point into consideration, which can extract the ground points more accurately and facilitate the subsequent steps.
2) We propose a new definition of curvature. By analyzing the covariance matrix of the point's neighborhoods, the points are classified and the curvature is calculated, which can extract more discriminative features.

Manuscript received 8 June 2022; accepted 21 August 2022. Date of publication 25 August 2022; date of current version 13 October 2022. This letter was recommended for publication by Associate Editor J. Zhang and Editor J. Civera upon evaluation of the reviewers' comments. This work was supported by the National Key Research and Development Program of China under Grants 2021YFF0307900 and 2021YFF0306202. *(Corresponding author: Feng Gao.)*

The authors are with the State Key Laboratory of Mechanical System and Vibration, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: zlwang_97@sjtu.edu.cn; ylm20159@sjtu.edu.cn; fengg@sjtu.edu.cn; coolrain@sjtu.edu.cn).

Digital Object Identifier 10.1109/LRA.2022.3201689

3) In lidar mapping, we add two new residuals, vertical residual and pitch residual, to the objective function. These residuals will help to converge to the optimal solution in the vertical direction. Then we propose a two-step Levenberg-Marquardt method to solve the pose transformation.

This letter is organized as follows: Section II reviews the related work of lidar SLAM approaches. Section III describes the details of the proposed method, including ground segmentation, feature extraction, lidar odometry, and lidar mapping. Section IV shows experiment results, and Section V is the conclusion.

II. Related Work

In recent years, lidar SLAM has become a popular research topic since it is less affected by the environment and has higher accuracy. The most essential step in lidar SLAM is the matching of point clouds, which is treated as a scan registration problem in the literature. Iterative Closed Point (ICP) algorithm [9] is a typical solution. By iteratively minimizing the distance residual between point clouds, the pose transformation converges to the optimal solution. On this basis, many variants have been proposed [11], [12], [13]. [11] proposed an ICP variant that uses a point-to-line metric and an exact closed-form for minimizing such metric, which speeds up the iteration. [13] proposed a continuous-time trajectory estimation (CICP) method, which estimates a continuous-time trajectory that takes into account inter-sample pose errors. The Normal Distribution Transform (NDT) algorithm [14] is another typical solution. It subdivides space into cells and assigns a normal distribution to each cell. Different from ICP, it avoids the point-to-point correspondence problem in point cloud registration.

In order to improve computational efficiency, feature-based matching methods have attracted more attention. Lidar Odometry And Mapping (LOAM) [10] is a typical solution, which no longer matches the entire point cloud, but extracts feature points first, and implements lidar odometry by minimizing the point-to-line and point-to-plane distances. In addition, it divides motion estimation into two modules that operate independently at high frequency and low frequency, and it fuses their results to obtain the final pose estimation. On this basis, LeGO-LOAM [15] performed ground optimization and point cloud segmentation, thus it can achieve real-time pose estimation on a low-power embedded system. [16] proposed a real-time and low-drift LiDAR SLAM system using planes as the landmark for the indoor environment since planes ubiquitously exist in the indoor environment. [17] proposed a feature-based SLAM algorithm using 2D image projections of the 3D lidar point cloud, which use a camera parameters matrix to rasterize the 3D point cloud to an image. Then ORB feature detector is applied to these images. [18] proposed a framework for odometry, mapping, and ground segmentation using a backpack LiDAR system that achieves both real-time and low-drift performance. [19] proposed a novel optimized lidar odometry and mapping method using ground plane constraints and SegMatch-based loop detection, and the ground plane constraints were used to reduce matching errors.

TABLE I
Coordinate Systems and Notations

$\{L\}$	Lidar coordinate system	$\{W\}$	World coordinate system
$\mathcal{E}_k^L$	Corner points set	$e_{(i,k)}^L$	The i th point in $\mathcal{E}_k^L$
$\mathcal{H}_k^L$	Plane points set	$h_{(i,k)}^L$	The i th point in $\mathcal{H}_k^L$
$\mathcal{G}_k^L$	Ground points set	$g_{(i,k)}^L$	The i th point in $\mathcal{G}_k^L$
$\mathcal{E}_k^W$	Local corner map	$\mathcal{H}_k^W$	Local plane map
$f_{\mathcal{A}}^L$	Residual function $\mathcal{A}$ in lidar odometry	$f_{\mathcal{A}}^W$	Residual function $\mathcal{A}$ in lidar mapping
T_{k-1}^k	Transformation matrix from frame $k-1$ to frame k	T_k^W	Transformation matrix of frame k in $\{W\}$

Some other works improve performance through multi-sensor fusion. [20] proposed a lightweight lidar-inertial state estimator, which designs an iterated error-state Kalman filter to fuse a 6-axis IMU and a 3D lidar in a tightly-coupled scheme. [21] proposed a tightly coupled lidar-inertia odometry and mapping scheme, with the ability to incorporate multiple lidars with a complementary field of view. [22] proposed a multi-sensor fusion framework, which fuses measurement from lidar, inertial sensor, and visual camera. It is composed of filter-based odometry and factor graph optimization. Shan et al [23] proposed a framework for tightly-coupled lidar inertial odometry, which formulates lidar-inertial odometry atop a factor graph. Additional sensor measurements such as GPS can easily be incorporated into the framework as new factors.

III. Methodology

A. System Overview

Fig. 2 shows our proposed software system. It can be divided into *Scan Registration* and *Odometry and Mapping*. In *Scan Registration*, let $\mathcal{P}_k$ be the input point cloud of the lidar frame k, the *Ground Segmentation* module divides $\mathcal{P}_k$ into non-ground points set and ground points set $\mathcal{G}_k$ firstly. Then the non-ground points set is sent to the feature extraction module to extract the corner points set $\mathcal{E}_k$ and plane points set $\mathcal{H}_k$. In *Odometry and Mapping*, two algorithms are running at the same time. *Lidar Odometry* provides high-frequency pose estimation T_k^{k-1}, while *Lidar Mapping* provides low-frequency but more accurate pose estimation T_k^W and registers the point cloud on the global map. Finally, the final pose estimation is output by the fusion of the two pose estimates through the *Transform Integration* module. The details of these modules are introduced below. For the convenience of reading, we summarize the involved coordinate systems and notations in Table I.

B. Ground Segmentation

Ground segmentation is a significant module in scan registration, since ground points set $\mathcal{G}_k$ does not need feature extraction, which reduces the calculating burden. Ground segmentation is divided into two steps: coarse segmentation and fine segmentation.

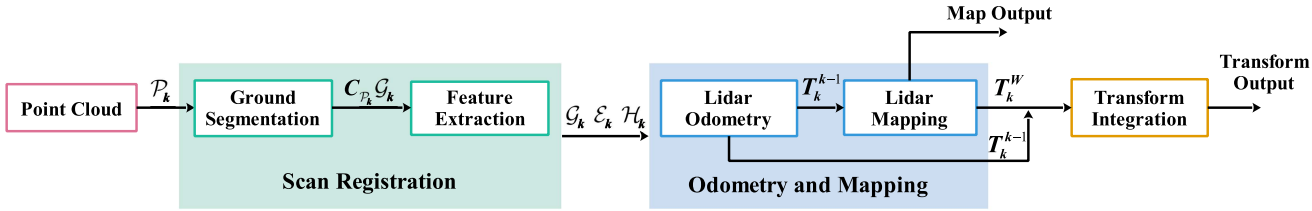


Fig. 2. Schematic diagram of our proposed software system.

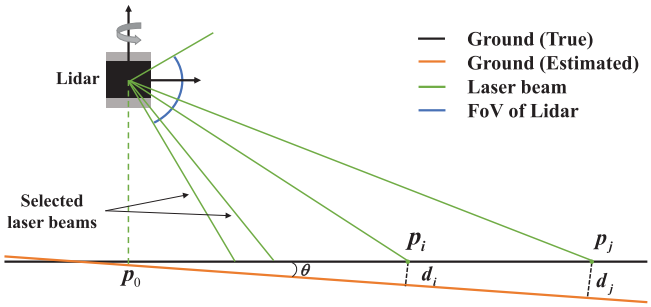
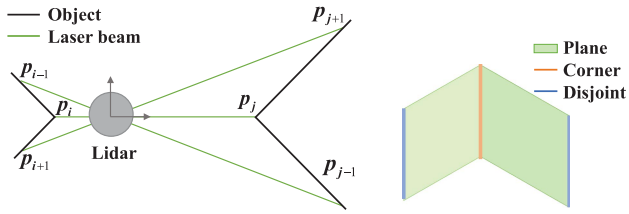


Fig. 3. Coarse ground segmentation.



(a) Defects of the LOAM-based method. (b) Feature types.

Fig. 4. Illustration of feature extraction.

粗分割：选取垂直角度最小的m条波束，用RANSAC计算地平面方程

It is assumed that there is no sharp height or angle change on the ground. This is acceptable since most working scenes of autonomous vehicles or mobile robots meet this assumption. Coarse segmentation is achieved in two steps. Firstly, we estimate a coarse ground plane equation. We do not estimate the ground equation directly based on the distance from the lidar to the ground, since the ground is not necessarily flat, and the lidar cannot be absolutely assembled horizontally. As shown in Fig. 3, in the coarse segmentation, we select m effective laser beams with the smallest vertical angle. They must be ground points if we filter out the laser beams hitting the autonomous vehicle or robot in advance. Then, we apply RANSAC [24] to the selected point cloud and estimate the ground plane equation $A_k x + B_k y + C_k z + D_k = 0$. Generally, m should not be too large. According to the number of lines of the lidar, $m \leqslant 3$ can be used, in order to reduce the calculating burden while ensuring that most of these points are ground points.

Secondly, we use the relationship between the points and the estimated plane equation to segment the ground points. Let $\mathcal{P}_k = \{p_1, p_2, \ldots, p_n\}$. Since we only select part of the points in the ground point set $\mathcal{G}_k$ to estimate the ground equation, it is inevitable to produce errors. Thus, there is an included angle θ between the estimated plane and the real ground plane, as shown in Fig. 3. This makes it impossible for us to simply use the point-to-plane distance to determine whether point p_i is a ground point, because the distances from points at different positions to the plane may vary greatly (d_i and d_j in the figure). Therefore, we use the following equation to determine whether point p_i is a ground point:

判断pi是否为地面点

$$\frac{d_i}{\|p_i - p_0\|_2} \leqslant |sin(\theta_{threshold})| \tag{1}$$

我觉得不直接精分割的原因肯定是计算量大，先用粗分割筛选掉一部分，精分割的计算量我觉得最有可能体现在KD-tree计算点的法这一步

Where d_i is the distance from point p_i to the estimated plane, $p_0 = (0, 0, -D_k/C_k)$, $\theta_{threshold}$ is the set threshold. As long as the above equation is established, the point p_i can be roughly regarded as the ground point. This ensures that distant ground points like p_j can also be included.

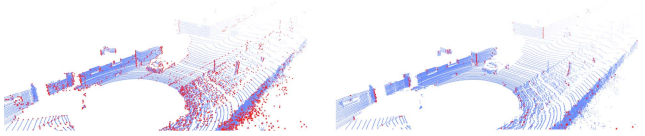
For fine segmentation, we build a 3D KD-tree [25] for the ground points extracted in the coarse segmentation and estimate the normal direction n_i of each point. If absolute value of the angle between n_i and $n_G = (A_k, B_k, C_k)$ is less than a threshold, we consider that the point is indeed a ground point. This filters out erroneous ground points in the coarse segmentation.

C. Feature Extraction

精分割：计算粗分割中地面点的法线，如果法线与地平面方程法线夹角绝对值小于阈值则认为确实为地面点

In both LOAM and LeGO-LOAM, the curvature of a point is calculated by the average distance between the point and its adjacent points on the same laser beam. If the distance is less than the threshold, the point is classified as a plane point, otherwise, it is a corner point. However, as shown in Fig. 4(a), it is obvious that the curvature of point p_i and point p_j should be the same in theory, but due to the different distances between them and the lidar, the LOAM-based method will get different curvature values, since $\|p_i - p_{i-1}\|_2 + \|p_i - p_{i+1}\|_2 \neq \|p_j - p_{j-1}\|_2 + \|p_j - p_{j+1}\|_2$. In addition, due to external environmental factors, such as object surface reflectivity, ambient light interference, and so on, the sparsity of point clouds at different positions is also different, which will also affect the LOAM-based curvature calculation. Therefore we propose a new curvature calculation method.

As shown in Fig. 4(b), similar to [18], non-ground points can be divided into three categories, disjoint points, plane points, and corner points. Disjoint points are usually caused by object occlusion or object surface discontinuity. These points may be unstable. For example, if the occlusion is an irregular object,



(a) LOAM-based. (b) Our method.

Fig. 5. Comparison of feature extraction results. The blue points are the plane points and the red points are the corner points.

the edge points hit on the occlusion at different times do not correspond to the same position when the lidar is moving. In LOAM they are usually classified as corner points, while we think a small number of good corner points are better than a large number of ordinary corner points, thus we directly classify the disjoint points as plane points. We use the following parameter to determine whether the point p_i is disjoint:

$$r_{co}^{i} = max\left(\frac{\|p_{i+1}-p_i\|_2}{\|p_{i-1}-p_i\|_2}, \frac{\|p_{i-1}-p_i\|_2}{\|p_{i+1}-p_i\|_2}\right) \tag{2}$$

If r_{co}^{i} is less than the threshold, it is considered that p_i is a continuous point, otherwise, it is a disjoint point, and it will be directly classified as a plane point. These disjoint points will not be used for subsequent curvature calculations. For continuous points, we further judge whether it is a plane point or a corner point. As can be seen from Fig. 4(b), for a certain laser beam, a good corner point is usually formed by the intersection of two straight lines composed of plane points, and their included angle is within a certain range. Therefore, for point p_i, we select $2n$ nearest neighbor points on the same laser beam to form a back point set $\mathcal{L}_b^i = \{p_{i-n}, \ldots, p_i\}$ and a forward point set $\mathcal{L}_f^i = \{p_i, \ldots, p_{i+n}\}$, and use the following equation to calculate the covariance matrix Σ_b^i and Σ_f^i of the two point sets respectively:

$$\Sigma = \frac{1}{n+1}\sum_{\tilde{p}\in\mathcal{L}}(\tilde{p}-\mu)(\tilde{p}-\mu)^T, \mu = \frac{1}{n+1}\sum_{\tilde{p}\in\mathcal{L}}\tilde{p} \tag{3}$$

We define curvature C_i:

相当于左右两边两条线向量，夹角后取对数

$$C_i = \frac{\left|V_b^i\right| \times \left|V_f^i\right|}{\left|V_b^i \cdot V_f^i\right|} \tag{4}$$

Where V_b^i and V_f^i are the eigenvectors corresponding to the maximum eigenvalues of the covariance matrices Σ_b^i and Σ_f^i, respectively. Curvature C_i can be regarded as the absolute value of the cosine of the angle between V_b^i and V_f^i, and then take the reciprocal. The larger the curvature, the closer the included angle is to 90 degrees. The points with curvature greater than the threshold are classified as corner points, and the rest are plane points. It is worth mentioning that if the maximum eigenvalue of the covariance matrix Σ_b^i or Σ_f^i is not significantly greater than the other two eigenvalues, it indicates that there are no two good line features near the point, and such points will also be directly classified as plane points.

Fig. 5 shows the comparison of our method and LOAM-based method [10]. It can be seen that in LOAM-based method, many misclassified corners are mixed in the plane, while our method is more rigorous and accurate, which can extract more discriminative features.

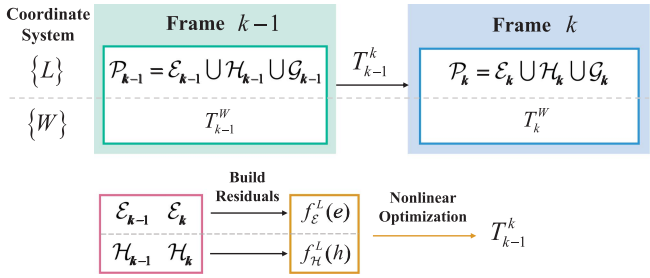


Fig. 6. Schematic diagram of lidar odometry.

D. Lidar Odometry

Lidar odometry needs to compute a high-frequency pose, which is based on scan-to-scan point cloud matching. Fig. 6 is the schematic diagram of lidar odometry. Let the pose in the world coordinate system corresponding to the point cloud of the $k-1$ th frame be $T_{k-1}^{W} \in SE(3)$, when we receive the point cloud $\mathcal{P}_k$ of frame k, the goal is to calculate the pose transformation $T_k^{k-1} \in SE(3)$ between the two point clouds and further calculate T_k^W. Similar to LOAM, we assume that the lidar moves with constant angular and linear velocities during the scanning process. Therefore, we first remove the motion distortion of the point cloud by linear interpolation. Then we estimate the pose based on the correspondence of the feature points. For $\mathcal{P}_k$, in the lidar coordinate system $\{L\}$, recall that the set of corner points $\mathcal{E}_k^L$ obtained by the feature extraction, and it is stored in a 3D KD-tree [25] for fast index. Other nearest neighbor searches below are also implemented through KD-tree. We define the residual term:

$$f_{\mathcal{E}}^{L}(e_{(i,k)}^{L}) = \frac{\left|(\tilde{e}_{(i,k)}^{L} - e_{(j,k-1)}^{L}) \times (\tilde{e}_{(i,k)}^{L} - e_{(l,k-1)}^{L})\right|}{\left|e_{(j,k-1)}^{L} - e_{(l,k-1)}^{L}\right|} \tag{5}$$

Where $\tilde{e}_{(i,k)}^{L} = T_k^{k-1} e_{(i,k)}^{L}, e_{(i,k)}^{L} \in \mathcal{E}_k^L, e_{(j,k-1)}^{L}$ is the nearest neighbor of $\tilde{e}_{(i,k)}^{L}$ in point set $\mathcal{E}_{k-1}^{L}$ on a same laser beam, and $e_{(l,k-1)}^{L}$ is the nearest neighbor on a different laser beam. The residual represents the point-to-line distance. Note that the constraint on the points' beam is to find a better residual. For example, the above constraints can ensure that points $\tilde{e}_{(i,k)}^{L}$, $e_{(j,k-1)}^{L}$, and $e_{(l,k-1)}^{L}$ are not collinear. For the plane points set $\mathcal{H}_k$, we define the residual term:

$$f_{\mathcal{H}}^{L}(h_{(i,k)}^{L}) = (\tilde{h}_{(i,k)}^{L} - h_{(j,k-1)}^{L}) \cdot n_{(i,k)}^{L} \tag{6}$$

$n_{(i,k)}^{L}$ is the unit vector given by

$$n_{(i,k)}^{L} = \frac{(h_{(j,k-1)}^{L} - h_{(m,k-1)}^{L}) \times (h_{(j,k-1)}^{L} - h_{(l,k-1)}^{L})}{\left|(h_{(j,k-1)}^{L} - h_{(m,k-1)}^{L}) \times (h_{(j,k-1)}^{L} - h_{(l,k-1)}^{L})\right|} \tag{7}$$

Where $\tilde{h}_{(i,k)}^{L} = T_k^{k-1} h_{(i,k)}^{L}, h_{(i,k)}^{L} \in \mathcal{H}_k^L$, $h_{(j,k-1)}^{L}$ and $h_{(l,k-1)}^{L}$ are the two nearest neighbors on the same laser beam

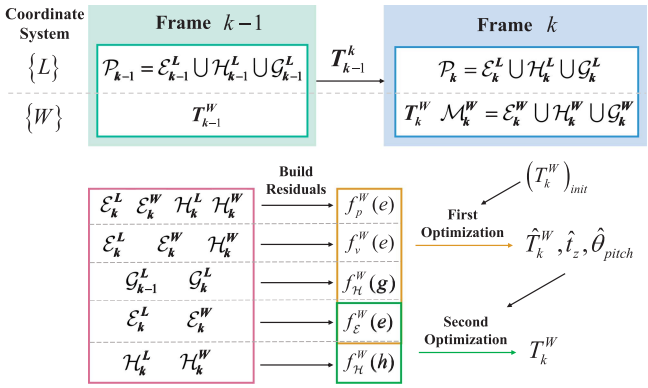


Fig. 7. Schematic diagram of lidar mapping.

of $\tilde{h}^L_{(i,k)}$, and $h^L_{(m,k-1)}$ is the nearest neighbor on a different laser beam. The residual represents the point-to-plane distance. Since we select more discriminative corner points in the feature extraction part, we give greater weight to the residuals of the corners. The pose is estimated by minimizing the weighted sum of the distance:

$$\min_{T^{k-1}_k} \sum f^L_{\mathcal{H}}(h^L_k) + W_{\mathcal{E}} \sum f^L_{\mathcal{E}}(e^L_k) \tag{8}$$

Where $W_{\mathcal{E}}$ is the weight of corner residual. The optimal pose estimation can be derived by the Levenberg-Marquardt method [26]:

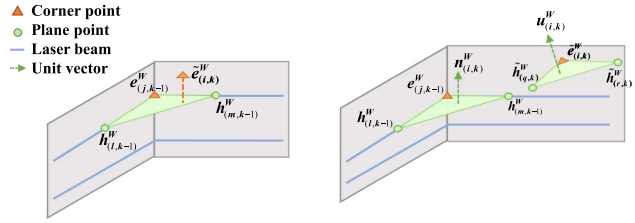
$$T^{k-1}_k \leftarrow T^{k-1}_k - (J^T J + \lambda diag(J^T J))^{-1} J^T f \tag{9}$$

Where λ is the Lagrange multiplier, J is the Jacobian matrix and f is the residual vector. To solve the nonlinear optimization, equation (9) is used to iterate and minimize the sum of the residuals in equation (8). The iteration can be stopped when the difference between the results of the two iterations is small or the maximum number of iterations is reached.

E. Lidar Mapping

Lidar mapping calculates a low-frequency but more accurate pose, and registers the point cloud to the global map. Fig. 7 is the schematic diagram of lidar mapping. The specific process is as follows. In the world coordinate system $\{W\}$, we maintain a global map $\mathcal{M}$, which consists of a corner point map $\mathcal{E}$, a plane point map $\mathcal{H}$, and a ground point map $\mathcal{G}$. That is, $\mathcal{M} = \mathcal{E} \cup \mathcal{H} \cup \mathcal{G}$. We define the pose of the $k-1$ th point cloud $\mathcal{P}_{k-1}$ in the world coordinate system as $T^W_{k-1} \in SE(3)$. At the end of sweep k, let $\mathcal{P}_k$ be the point cloud with the distortion removed, then the initial estimate of $\mathcal{P}_k$ is $(T^W_k)_{init} = T^k_{k-1} T^W_{k-1}$, where T^k_{k-1} is obtained by lidar odometry. In order to reduce the amount of computation, similar to the idea of sliding window, we extract the local map $\mathcal{M}^W_k = \mathcal{E}^W_k \cup \mathcal{H}^W_k \cup \mathcal{G}^W_k$ based on this initial estimate, and match $\mathcal{P}_k$ and $\mathcal{M}^W_k$ to obtain an accurate pose estimate T^W_k. Finally, according to this pose, the point cloud is transformed and added to the global map.

In LOAM, lidar mapping is similar to lidar odometry, which optimizes the pose by minimizing the residuals of corner and plane points. However, through experiments, we found that this



(a) Vertical residual. (b) Pitch residual.

Fig. 8. Schematic diagram of calculating residuals.

method will produce large errors in the vertical direction. We believe that the error is caused by two parts. First, the residual constraints of corner and plane points are not sensitive to vertical pose changes. Many extracted line features and plane features are nearly perpendicular to the horizontal plane, especially in an environment with dense regular buildings. Such features will make it difficult for the vertical translation to converge to the optimal solution, since the distances from points of different heights to the vertical line or plane are the same. In addition, we find that the inaccurate estimation of pitch angle will also cause vertical error. Especially when the forward distance is long, a small pitch angle error will cause a large vertical error. Based on the above analysis, we have made the following improvements to the lidar mapping.

Our lidar mapping is optimized in two steps. In the first step, we mainly estimate the vertical transformation and pitch angle transformation. We add the following residual, which is mainly used to estimate the vertical transformation:

$$f^W_v\left(e^L_{(i,k)}\right) = \left(\tilde{e}^W_{(i,k)} - e^W_{(j,k-1)}\right) \cdot n^W_{(i,k)} \tag{10}$$

$n^W_{(i,k)}$ is the unit vector perpendicular to the plane determined by $e^W_{(j,k-1)}, h^W_{(l,k-1)}$ and $h^W_{(m,k-1)}$:

$$n^W_{(i,k)} = \frac{\left(e^W_{(j,k-1)} - h^W_{(m,k-1)}\right) \times \left(e^W_{(j,k-1)} - h^W_{(l,k-1)}\right)}{\left|\left(e^W_{(j,k-1)} - h^W_{(m,k-1)}\right) \times \left(e^W_{(j,k-1)} - h^W_{(l,k-1)}\right)\right|} \tag{11}$$

Where $\tilde{e}^W_{(i,k)} = \hat{T}^W_k e^L_{(i,k)}, e^L_{(i,k)} \in \mathcal{E}^L_k$. $e^W_{(j,k-1)}$ is the nearest neighbor of $\tilde{e}^W_{(i,k)}$ in point set $\mathcal{E}^W_k$. $h^W_{(m,k-1)}$ and $h^W_{(l,k-1)}$ are the two nearest neighbors of $\tilde{e}^W_{(i,k)}$ in point set $\mathcal{H}^W_k$. These three points need to be guaranteed to be on the same laser beam. The residual represents the point-to-plane distance, as shown in Fig. 8(a). In addition, for each residual, we will judge whether the plane is approximately parallel to the ground. If not, the residual will not be added to the optimization function. The pitch angle transformation is mainly estimated by the following residual:

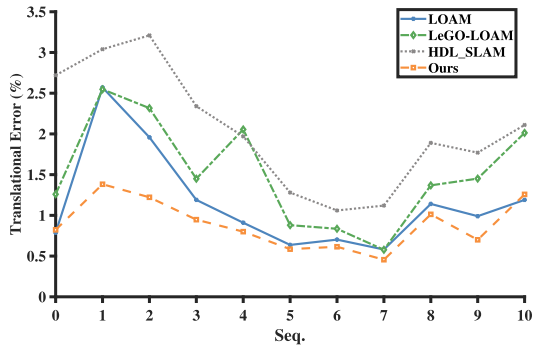
$$f^W_p(e^L_{(i,k)}) = \left| n^W_{(i,k)} \times u^W_{(i,k)} \right| \tag{12}$$

Where $u^W_{(i,k)}$ is the unit vector perpendicular to the plane determined by $\tilde{e}^W_{(i,k)}, \tilde{h}^W_{(q,k)}$ and $\tilde{h}^W_{(r,k)}$, similar to (11). $\tilde{h}^W_{(q,k)} = \hat{T}^W_k h^L_{(q,k)}, \tilde{h}^W_{(r,k)} = \hat{T}^W_k h^L_{(r,k)}$. $h^L_{(q,k)}$ and $h^L_{(r,k)}$ are the two nearest neighbors of $e^L_{(i,k)}$ in point set $\mathcal{H}^L_k$. These six points need to be guaranteed to be on the same laser beam. This residual

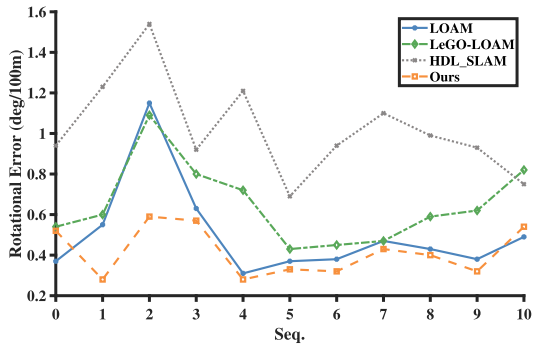
TABLE II
COMPARISON OF AVERAGE ERROR AND COMPUTING TIME OF DIFFERENT APPROACHES ON KITTI DATASET

Method	Average Error		Computing Time
	Translational(%)	Rotational(deg/100m)	(ms/frame)
LOAM	1.1504	0.5027	60.57
LeGO-LOAM	1.5230	0.6481	**41.51**
HDL-SLAM	2.0461	1.0225	105.27
Ours	**0.8908**	**0.4163**	58.37
Ours-NV	0.9787	0.4485	52.10
Ours-NP	1.0265	0.4572	55.44
Ours-NVP	1.0589	0.4820	49.28

Ours-NV: Our method without vertical residual.
Ours-NP: Our method without pitch residual.
Ours-NVP: Our method without vertical residual and pitch residual.



(a) Translational Error.



(b) Rotational Error.

Fig. 9. Comparison of the different approaches on KITTI dataset sequence 00-10. The errors are measured using segments of trajectories at 100, 200,..., 800 m lengths based on 3D coordinates, and the figure shows the average value.

represents the sine of the angle between the normal vectors of the two planes, as show in Fig. 8(b).

In addition, we also added the residuals of corner and ground points to the optimization function. The corner residual can constrain the horizontal transformation, while the previous two residuals cannot. Similar to (5), the corner points $\mathcal{E}_k^L$ are matched with the local corner map $\mathcal{E}_k^W$. Ground points are mainly used to estimate the transformation of angle. Note that in order to reduce computing expense, the ground points are matched with the point cloud of $k-1$ th frame, not the local map. So far, we

TABLE III
COMPARISON OF AVERAGE VERTICAL TRANSLATION ERROR OF DIFFERENT APPROACHES ON KITTI DATASET

Method	LOAM	LeGO-LOAM	HDL-SLAM	Ours
Vertical Error (%)	0.3688	0.3389	0.4774	**0.1557**

define the optimization function of the first step:

$$\min_{\hat{T}_k^W} \sum f_{\mathcal{E}}^W(e_k^L) + W_{\mathcal{G}} \sum f_{\mathcal{H}}^W(g_k^L) + W_v \sum f_v^W(e_k^L) + W_p \sum f_p^W(e_k^L) \tag{13}$$

Where $g_k^L \in \mathcal{G}_k^L$, $W_{\mathcal{G}}, W_v$ and W_p are the weights. The main purpose of the first step optimization is to calculate the estimated vertical transformation $\hat{t}_z$ and the pitch angle $\hat{\theta}_{pitch}$ in $\hat{T}_k^W$. They will be used in the second step.

In the second step, we estimate an accurate T_k^W. We match the corner points $\mathcal{E}_k^L$ and plane points $\mathcal{H}_k^L$ with the local map and add the constraints of vertical transformation and pitch angle. Different from LeGO-LOAM, we do not take the $\hat{t}_z$ and $\hat{\theta}_{pitch}$ estimated in the first step as the output directly, since their errors may lead to the subsequent pose estimation being not robust. Similar to optimization problems with inequality constraints, we put constraints into the objective function:

$$\min_{T_k^W} \sum f_{\mathcal{H}}^W(h_k^L) + W_{\mathcal{E}} \sum f_{\mathcal{E}}^W(e_k^L) + W_z \left\| t_z - \hat{t}_z \right\|_2 + W_{pi} \left\| \theta_{pitch} - \hat{\theta}_{pitch} \right\|_2 \tag{14}$$

Where t_z and θ_{pitch} are the vertical transformation and pitch angle in T_k^W, respectively. (13) and (14) can be derived by the Levenberg-Marquardt method, and the initial value is $(T_k^W)_{init}$. Note that the weights in equation (13) and equation (14) are empirical values, which can be determined through experiments. For example, if we use a lidar with 64 laser beams, we can set $W_{\mathcal{G}} = 0.5, W_v = 50, W_p = 80, W_{\mathcal{E}} = 7, W_z = 2000, W_{pi} = 1400$. They can be adjusted according to the model and working environment of the lidar. Finally, the point cloud after pose transformation is downsized by voxel filtering and then registered in the global map $\mathcal{M}$. For transform integration, we fuse the poses from the lidar odometry and the lidar mapping, at the same frequency as the lidar odometry.

IV. EXPERIMENTS

In this section, we verify the performance of our proposed method through experiments on public datasets and real environments. Additionally, we compare our method with state-of-the-art methods. All algorithms are implemented in C++ and executed using the robot operating system (ROS) [27] in Ubuntu Linux.

A. Experiments on Public Dataset

To verify the effectiveness of our proposed method, we conduct experiments on the KITTI odometry dataset [28], which

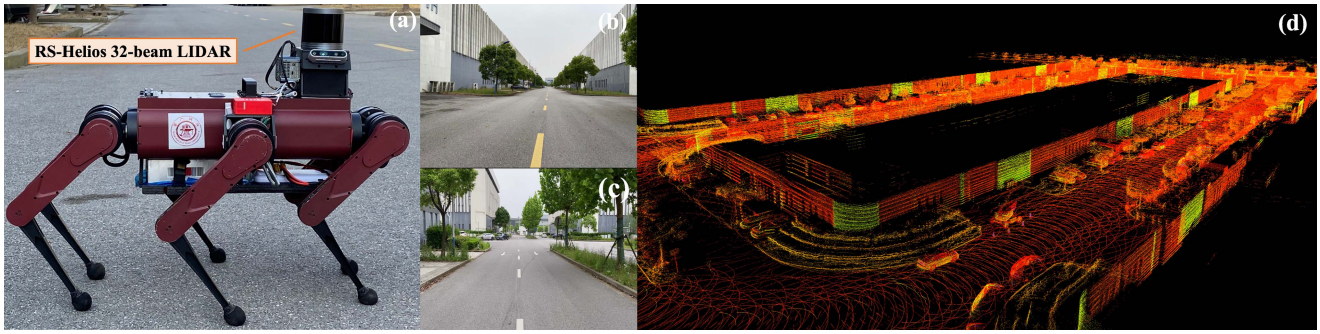


Fig. 10. FeVO-LOAM in real environment. (a) Hexapod robot used for experiment. (b)–(c) The industrial plant. (d) Mapping result.

TABLE IV
RMSE TRANSLATION ERROR W.R.T LIO-SAM

Method	LOAM	LeGO-LOAM	HDL-SLAM	Ours
RMSE (m)	5.0409	2.9032	3.4400	**0.1636**

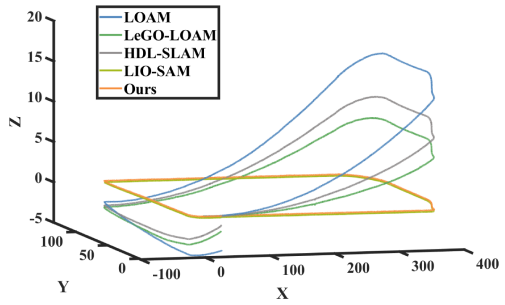


Fig. 11. Trajectories generated by different methods in the real-world experiment.

is popularly used for the evaluation of lidar odometry. In addition, we adopt the evaluation metrics in [28] to calculate the translation and rotation errors respectively:

$$E_{trans}(\mathcal{F}) = \frac{1}{|\mathcal{F}|} \sum_{(k,s)\in\mathcal{F}} \left\| \hat{T}_s \hat{T}_k^{-1} T_k T_s^{-1} \right\|_2$$

$$E_{rot}(\mathcal{F}) = \frac{1}{|\mathcal{F}|} \sum_{(k,s)\in\mathcal{F}} \angle \left[\hat{T}_s \hat{T}_k^{-1} T_k T_s^{-1} \right] \tag{15}$$

Where $\hat{T}, T \in SE(3)$ are the estimated and true poses respectively, $\mathcal{F}$ is a set of frames (k, s), $\angle[\cdot]$ is the rotation angle. We compare our algorithm with the open-sourced state-of-the-art lidar SLAM algorithms, including LOAM [10], LeGO-LOAM [15], and HDL-Graph-SLAM [29]. All algorithms are tested on an Intel i7 2.3 GHz processor-based computer. The results are shown in Table II and Fig. 9.

It can be seen that our method achieves the highest accuracy in 9 of the 11 sequences, and the average error of our method is minimal. This is mainly because our method can greatly reduce the error of vertical direction and pitch angle. In terms of computing time, LeGO-LOAM is the fastest approach, while our method also achieves an average processing rate of more than 10 Hz, which is a good trade-off between computational cost and localization accuracy.

In addition, to study the importance of vertical residual and pitch residual in lidar mapping, we conduct an ablation study on our method by removing different residuals, and the results are shown in Table II. Obviously, our method does not achieve the best results when these residuals are removed, which demonstrates the importance of the proposed residuals. Note that Ours-NVP removes the two proposed residuals, and it simplifies lidar mapping to a one-step Levenberg-Marquardt optimization, which is the same as that of LOAM. Thus its main difference from LOAM is the feature extraction module. It can be seen that its result is better than LOAM, which also proves the superiority of our feature extraction module. The results of ablation experiments show that each step we propose is indispensable. The proposed feature extraction module is used to extract more discriminative feature points, and the proposed new residuals are used to estimate a more accurately vertical pose transformation. Among them, the pitch residual contributes the most to the overall improvement, since inaccurate pitch angle estimation will lead to a large cumulative drift.

Moreover, we compare the vertical translation error of all methods, and its calculation method is similar to equation (15). The results are shown in Table III. It can be seen that the vertical translation error of our method is 0.1557%. Compared with other methods, our method can reduce the vertical translation error by more than half.

B. Experiments in Real Environment

In this experiment, we apply our algorithm to a real robot. As shown in Fig. 10, the robot used in this experiment is a hexapod robot called SJTU-Hexapod-Mini. It has good terrain adaptability and motion performance, which can walk on a slope of 20 degrees, climb 15 cm high stairs, and reach a maximum forward speed of 2.1 m/s at the ground. The robot is equipped with an RS-Helios 32-beam LIDAR for localization. This robot is mainly used for blind guidance, exploration, and patrol inspection. All these functions require precise localization to ensure safety and reliability.

The proposed method is tested in an outdoor real environment to evaluate its mapping and localization accuracy, as shown in Fig. 10. The environment is a common industrial plant, and the robot is remotely controlled to move around the test area. The robot was driven at an average speed of 1.2 m/s. Finally, we let the robot return to the initial position to verify whether the

closed loop can be completed. The total length of the path is about 1.2 km. The elevation changes in this environment are small, but the vibration of the robot will bring challenges to the localization. The whole experiment was implemented on an Nvidia Jetson AGX computer equipped with this robot. We also conduct localization experiments using other state-of-the-art methods on the same device. In addition, since it is difficult to collect GPS data in this environment, we use the results of LIO-SAM [23] as the comparison standard. It is a SLAM framework for tightly-coupled lidar inertial odometry, which achieves highly accurate, real-time mobile robot trajectory estimation and map-building. The RMSE error is shown in Table IV. The comparison of trajectories is shown in Fig. 11.

It can be seen that our mapping result has high accuracy and can complete the loop closure without loop closure detection. The trajectory of other methods has a large upward drift, and they cannot complete the loop closure. This is mainly because buildings in the environment will produce line features and surface features nearly perpendicular to the horizontal plane, which makes it difficult for the vertical translation of LOAM-based approaches to converge to the optimal solution. This result also proves that our proposed approach can indeed reduce the vertical error.

V. Conclusion

In this letter, we propose an efficient and high-accuracy lidar SLAM framework, which targets to reduce the vertical error of lidar SLAM. Compared with the traditional LOAM-based method, we propose a new ground segmentation approach, which takes the distance from the point to the lidar and the normal direction of each point into consideration, thus it can extract the ground points more accurately and facilitate the subsequent steps. In addition, we propose a new definition of curvature, which can extract more discriminative features. Moreover, in lidar mapping, we add two new residuals, vertical residual and pitch residual, to the objective function. Then we use the two-step Levenberg-Marquardt method to solve the pose transformation. Experiments on the KITTI dataset and real environment show that our framework outperforms other state-of-the-art methods while consuming similar computational resources, and our framework can also significantly reduce vertical error. The future work will include the fusion with IMU and other sensors. It will bring more prior estimates to the pose transformation, thus obtaining a more accurate pose estimation.

References

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

[2] I. Vallivaara, J. Haverinen, A. Kemppainen, and J. Röning, "Magnetic field-based SLAM method for solving the localization problem in mobile robot floor-cleaning task," in *Proc. IEEE 15th Int. Conf. Adv. Robot. (ICAR)*, 2011, pp. 198–203.

[3] E. Westman and M. Kaess, "Degeneracy-aware imaging sonar simultaneous localization and mapping," *IEEE J. Ocean. Eng.*, vol. 45, no. 4, pp. 1280–1294, Oct. 2020.

[4] 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)*, 2021, pp. 4390–4396.

[5] C. Debeunne and D. Vivet, "A review of visual-LiDAR fusion based simultaneous localization and mapping," *Sensors*, vol. 20, no. 7, 2020, Art. no. 2068.

[6] M. Karimi, M. Oelsch, O. Stengel, E. Babaians, and E. Steinbach, "LoLa-SLAM: Low-latency LiDAR SLAM using continuous scan slicing," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 2248–2255, Apr. 2021.

[7] S. Cho, C. Kim, J. Park, M. Sunwoo, and K. Jo, "Semantic point cloud mapping of LiDAR based on probabilistic uncertainty modeling for autonomous driving," *Sensors*, vol. 20, no. 20, 2020, Art. no. 5900.

[8] M. Ramezani, G. Tinchev, E. Iuganov, and M. Fallon, "Online LiDAR-SLAM for legged robots with robust registration and deep-learned loop closure," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, 2020, pp. 4158–4164.

[9] P. J. Besl and N. D. McKay, "Method for registration of 3-D shapes," *Sensor Fusion IV: Control Paradigms Data Structures*, vol. 1611, pp. 586–606, 1992.

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

[11] A. Censi, "An ICP variant using a point-to-line metric," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2008, pp. 19–25.

[12] H. Kim, S. Song, and H. Myung, "GP-ICP: Ground plane ICP for mobile robots," *IEEE Access*, vol. 7, pp. 76599–76610, 2019.

[13] H. Alismail, L. D. Baker, and B. Browning, "Continuous trajectory estimation for 3D SLAM from actuated lidar," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, 2014, pp. 6096–6101.

[14] M. Magnusson, A. Lilienthal, and T. Duckett, "Scan registration for autonomous mining vehicles using 3D-NDT," *J. Field Robot.*, vol. 24, no. 10, pp. 803–827, 2007. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.20204

[15] 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)*, 2018, pp. 4758–4765.

[16] L. Zhou, D. Koppel, and M. Kaess, "LiDAR SLAM with plane adjustment for indoor environment," *IEEE Robot. Automat. Lett.*, vol. 6, no. 4, pp. 7073–7080, Oct. 2021.

[17] W. Ali, P. Liu, R. Ying, and Z. Gong, "A feature based laser SLAM using rasterized images of 3D point cloud," *IEEE Sensors J.*, vol. 21, no. 21, pp. 24422–24430, Nov. 2021.

[18] P. Chen, W. Shi, S. Bao, M. Wang, W. Fan, and H. Xiang, "Low-drift odometry, mapping and ground segmentation using a backpack LiDAR system," *IEEE Robot. Automat. Lett.*, vol. 6, no. 4, pp. 7285–7292, Oct. 2021.

[19] X. Liu, L. Zhang, S. Qin, D. Tian, S. Ouyang, and C. Chen, "Optimized LOAM using ground plane constraints and SegMatch-based loop detection," *Sensors*, vol. 19, no. 24, 2019, Art. no. 5419.

[20] C. Qin, H. Ye, C. E. Pranata, J. Han, S. Zhang, and M. Liu, "LINS: A lidar-inertial state estimator for robust and efficient navigation," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, 2020, pp. 8899–8906.

[21] T.-M. Nguyen, S. Yuan, M. Cao, L. Yang, T. H. Nguyen, and L. Xie, "MILIOM: Tightly coupled multi-input lidar-inertia odometry and mapping," *IEEE Robot. Automat. Lett.*, vol. 6, no. 3, pp. 5573–5580, Jul. 2021.

[22] J. Lin, C. Zheng, W. Xu, and F. Zhang, "R ^{2} LIVE: A robust, real-time, LiDAR-inertial-visual tightly-coupled state estimator and mapping," *IEEE Robot. Automat. Lett.*, vol. 6, no. 4, pp. 7469–7476, Oct. 2021.

[23] 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)*, 2020, pp. 5135–5142.

[24] R. Schnabel, R. Wahl, and R. Klein, "Efficient RANSAC for point-cloud shape detection," in *Comput. Graph. Forum*, vol. 26, no. 2, 2007, pp. 214–226.

[25] B. D. Mark, C. Otfried, v. K. Mark, and O. Mark, *Computational Geometry Algorithms and Applications*. Berlin, Germany: Spinger, 2008.

[26] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge, U.K.: Cambridge Univ. Press, 2003.

[27] M. Quigley et al., "ROS: An open-source Robot Operating System," in *Proc. ICRA Workshop Open Source Softw.*, 2009, vol. 3,pp. 1–6.

[28] 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.*, 2012, pp. 3354–3361.

[29] 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. Robotic Syst.*, vol. 16, no. 2, 2019, Art. no. 1729881419841532.