Robotica (2025), **43**, pp. 2872–2891
doi:10.1017/S0263574725101926

CAMBRIDGE UNIVERSITY PRESS

RESEARCH ARTICLE

A multi-metric LiDAR Simultaneous Localization and Mapping system based on intensity feature assistance and two-step decoupling estimation

Zhenghui Xu[1], Jian Li[1], Shimin Wei[1], Ling Tang[1] and Huanlong Chen[2]

[1]School of Intelligent Engineering and Automation, Beijing University of Posts and Telecommunications, Beijing, China
[2]Shanghai Institute of Aerospace System Engineering, Shanghai, China
Corresponding author: Jian Li; Email: jianli_628@126.com

Received: 11 March 2025; **Revised:** 28 May 2025; **Accepted:** 5 June 2025; **First published online:** 22 July 2025

Keywords: LiDAR Simultaneous Localization and Mapping; multi-metric system; intensity feature; two-step decoupling

Abstract

To address the problems of accuracy degradation, localization drift, and even failure of Simultaneous Localization and Mapping (SLAM) algorithms in unstructured environments with sparse geometric features, such as outdoor parks, highways, and urban roads, a multi-metric light detection and ranging (LiDAR) SLAM system based on the fusion of geometric and intensity features is proposed. Firstly, an adaptive method for extracting multiple types of geometric features and salient intensity features is proposed to address the issue of insufficient sparse feature extraction. In addition to extracting traditional edge and planar features, vertex features are also extracted to fully utilize the geometric information, and intensity edge features are extracted in areas with significant intensity changes to increase multi-level perception of the environment. Secondly, in the state estimation, a multi-metric error estimation method based on point-to-point, point-to-line, and point-to-plane is used, and a two-step decoupling strategy is employed to enhance pose estimation accuracy. Finally, qualitative and quantitative experiments on public datasets demonstrate that compared to state-of-the-art pure geometric and intensity-assisted LiDAR SLAM algorithms, our proposed algorithm achieves superior localization accuracy and mapping clarity, with an ATE accuracy improvement of 28.93% and real-time performance of up to 62.9 ms. Additionally, test conducted in real campus environments further validates the effectiveness of our approach in complex, unstructured scenarios.

1. Introduction

With the rapid development of autonomous navigation technology for robots, Simultaneous Localization and Mapping (SLAM) has gradually become one of its key technologies [1, 2]. SLAM enables precise six-degree-of-freedom localization and dense three-dimensional mapping in unknown environments solely using onboard sensors, without relying on external localization systems such as GPS, making it an increasingly prominent research focus [3]. Throughout nearly four decades of SLAM development, current mainstream SLAM algorithms can be categorized into Visual SLAM (V-SLAM) [4] and light detection and ranging (LiDAR) SLAM [5] based on their primary sensing modalities. Visual SLAM primarily utilizes cameras for pose estimation and mapping, including monocular, stereo, and RGB-D cameras. While this approach offers advantages such as cost-effectiveness, portability, and extended measurement range, it heavily depends on environmental texture and is sensitive to illumination variations, resulting in reduced robustness and accuracy in degraded environments [6]. In contrast, LiDAR SLAM, which employs light detection and ranging, demonstrates superior robustness to variations in illumination, texture, and weather conditions [7–9]. Furthermore, its direct acquisition of point cloud data inherently enables high-precision pose estimation, leading to its widespread adoption in industrial applications.

© The Author(s), 2025. Published by Cambridge University Press.

https://doi.org/10.1017/S0263574725101926 Published online by Cambridge University Press

Traditional LiDAR SLAM algorithms [10–26] primarily rely on geometric feature extraction from the surrounding environment for point cloud registration. These methods distinguish edge and planar points through simple curvature calculations, demonstrating satisfactory performance in geometrically rich environments. However, they exhibit reduced accuracy in feature-sparse and degenerate scenarios. In contrast, LiDAR intensity information, an inherent component of measurement data, effectively captures the distinctive reflectivity characteristics of different surface materials. This additional modality can be leveraged to enhance pose estimation accuracy and improve robustness in degraded environments [27, 28].

Recently, several intensity-assisted LiDAR SLAM algorithms [29–33] have been proposed. However, these approaches simultaneously extract both geometric and intensity-based edge and planar features, resulting in substantial computational overhead and reduced real-time performance. Given that intensity data is one-dimensional, intensity planar features, which represent surface characteristics of identical materials, not only lack distinctiveness but also require significant computational resources. In contrast, intensity edge features, characterized by significant variations in intensity values, exhibit more distinguishable and easily extractable properties. Therefore, by focusing exclusively on intensity edge point extraction, more salient features can be obtained while reducing computational complexity.

Although the aforementioned algorithms improve the accuracy of pose estimation, they still exhibit limitations in unstructured environments, as these existing algorithms uniformly process geometric and intensity features extensively, which reduces the real-time performance of the algorithms. Additionally, they often rely solely on point-to-line or point-to-plane error metrics, lacking more refined joint error calculations, which leads to decreased localization accuracy in degraded scenarios.

To address the challenges of insufficient feature extraction and degraded localization accuracy in unstructured environments, we propose a novel multi-metric LiDAR SLAM system. This system enables the extraction of multiple types of geometric features and salient intensity features while performing multi-metric registration and pose decoupling to achieve precise, robust, and real-time localization and mapping. The main contributions of this article include:

1) We propose a multi-metric LiDAR SLAM system that performs multi-metric pose estimation by extracting diverse geometric and intensity features, thereby improving the accuracy and robustness of localization.
2) We use an adaptive feature extraction method that can not only extract multiple types of geometric features but also extract significant intensity edge features in areas of intensity variation to fully utilize environmental information.
3) We design a novel multi-weight function to fuse local geometric and intensity information, incorporating a two-step decoupling strategy to solve different pose components to improve the accuracy of pose estimation.
4) Extensive qualitative and quantitative evaluations on various public datasets and real-world scenarios demonstrate the accuracy, robustness, and practicality of our proposed method, particularly its effectiveness in unstructured environments.

The rest of this paper is organized as follows: Section 2 gives a review of relevant literature. Section 3 offers a comprehensive explanation of the proposed methodology. Section 4 showcases experimental results and performance evaluations. Finally, a brief discussion and conclusion are drawn in Section 5.

2. Related works

2.1. Traditional LiDAR SLAM

The early LiDAR SLAM utilized all raw measurement data for point cloud registration, employing two classic registration methods [10]: Iterative Closest Point (ICP) and Normal Distribution Transformation (NDT). The ICP method [11, 12], as the name suggests, matches each point in the current frame with

https://doi.org/10.1017/S0263574725101926 Published online by Cambridge University Press

the nearest point in the target frame through nearest neighbor search. ICP iteratively searches for the optimal point pairs by minimizing the Euclidean distance between point pairs until the transformation matrix converges. This method has been widely applied in LiDAR SLAM algorithms, such as SuMa [13], LiDAR-Odometry and Localization (LOL) [14], and CT-ICP [15]. On the other hand, NDT [16] is a mathematically-based method that models discrete point cloud data using Gaussian distributions, transforming the point cloud into a distribution function of points, thereby calculating the relative pose relationship between point clouds. Related algorithms include HDL-Graph SLAM [17], NDT-PSO [18], and Ndt-transformer [19].

Directly using all raw point cloud data can lead to longer processing times and is susceptible to measurement noise, resulting in lower accuracy and robustness. Therefore, researchers have further improved the methods by proposing feature-based matching techniques for point clouds. Zhang et al. [20] first introduced LOAM, which extracts edge and plane features by analyzing point cloud curvature. This approach divides the robot's motion estimation problem into two threads: odometry and mapping, ensuring real-time requirements. Shan et al. [21] proposed LeGO-LOAM, which utilizes ground point separation to cluster and label ground and non-ground points. It estimates roll, pitch, and z-axis positions using plane features, while edge features are used to estimate yaw, x-axis, and y-axis positions. The final transformation fusion achieves fast and high-precision localization. Wang et al. [22] introduced FLOAM, which employs a rapid two-stage distortion compensation to reduce computational costs and considers the smoothness of local features during pose optimization, thereby improving algorithm accuracy and real-time performance. To further enhance localization accuracy and algorithm versatility, Pan et al. [23] proposed MULLS, a pure LiDAR SLAM solution compatible with various LiDAR sensors. MULLS extracts multiple geometric features through Principal Component Analysis (PCA) [24] in the front-end, performs point cloud registration using a multi-metric error estimation and least squares iterative closest point algorithm, and implements pose graph optimization in the back-end. Chen et al. [25] presented NDT-LOAM, which is based on weighted Normal Distribution Transformation (NDT) and Local Feature Adjustment (LFA), ensuring real-time performance and high accuracy through improved weighted NDT, keyframes, and LFA. And Xu et al. [26] proposed using stable feature matching based on geometric feature sets and optimizing global pose drift through three-stage loop detection.

However, the aforementioned methods overlook the intensity information of LiDAR, leading to insufficient perception of effective environmental information, which makes them unable to effectively cope with feature-sparse degraded scenarios.

2.2. Intensity-assisted LiDAR SLAM

Currently, some researchers have utilized the intensity information from LiDAR as auxiliary features for point cloud registration, achieving better results. Park et al. [29] proposed I-LOAM, which incorporates intensity data into the feature cost function of LOAM, enhancing the accuracy and robustness of the algorithm. Wang et al. [30] introduced an intensity-assisted LiDAR SLAM (Amplitude Modulation SLAM), which effectively optimizes localization accuracy by utilizing geometric and intensity residuals to construct intensity maps. Li et al. [31] developed a solid-state LiDAR SLAM algorithm based on intensity and geometric feature extraction, designing two polynomial weighting functions specifically for planar and edge features. Dai et al. [32] proposed an adaptive intensity feature extraction and intensity constraint construction method based on degradation detection for registration optimization. Zhang et al. [33] introduced an adaptive selection strategy for intensity and geometric features, determining whether to select intensity features based on the richness of geometric features. Although this approach can improve pose estimation accuracy, it is not suitable for degraded scenarios.

These methods typically only perform error measurement from points to lines or points to planes, without a more refined multi-scale joint linear measurement from points to points, points to lines, and points to planes, resulting in lower localization accuracy in degraded scenarios. In this paper, we adopt a multi-scale error joint linear estimation and further introduce a two-step pose decoupling to enhance the accuracy of pose estimation.

https://doi.org/10.1017/S0263574725101926 Published online by Cambridge University Press

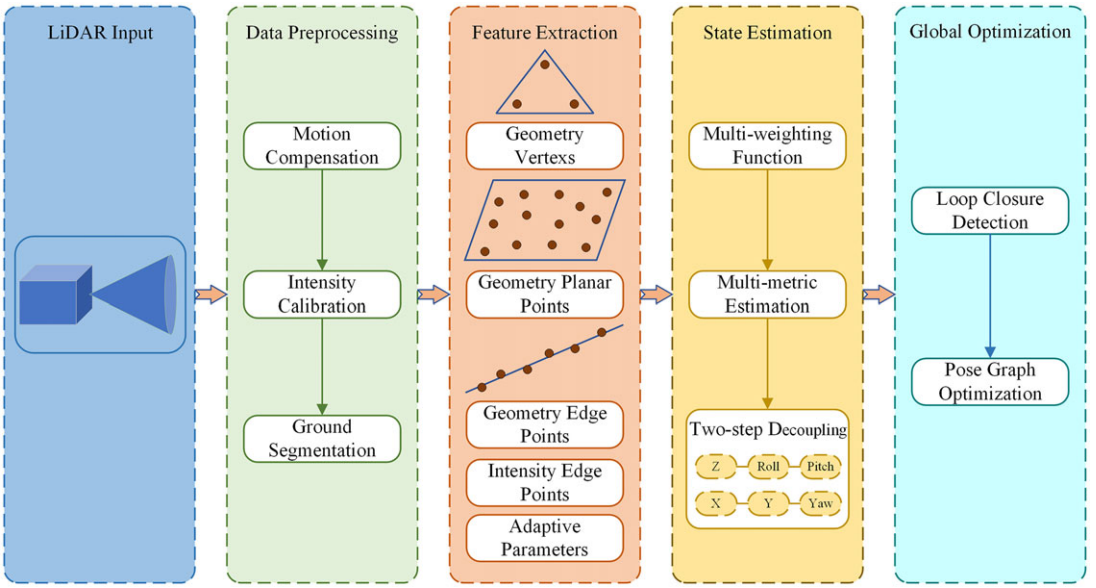


Figure 1. *Overview of our proposed system. Preprocessing is performed to compensate, correct, and separate the point cloud, followed by the adaptive extraction of multiple geometric features of vertices, edge points, and plane points, as well as intensity edge feature points. Subsequently, multi-metric feature matching and two-step decoupling are conducted, and finally, global optimization is performed.*

3. System overview

The pipeline of the proposed method is illustrated in Figure 1. The entire system consists of four parts: data preprocessing, feature extraction, state estimation, and global optimization. The preprocessing module performs motion distortion compensation for LiDAR frames, intensity data calibration, and ground point segmentation. The feature extraction module extracts multiple types of feature points from both geometric and intensity data. The state estimation module performs multi-metric registration of various extracted features coupled with a two-step decoupling strategy. Finally, the global optimization module executes loop closure detection and graph optimization.

3.1. Data preprocessing

3.1.1. Motion compensation

To address the motion distortion of point cloud frames, all points in the current or given frame are projected to either the start or end time of the frame. Assuming constant velocity motion within the frame, the transformation formula for projecting any point $\boldsymbol{p}_i$ to the frame's end time is given by:

$$\boldsymbol{p}_i' = slerp\left(\boldsymbol{R}_e, \boldsymbol{p}_e, s\right) + \boldsymbol{t}_e \tag{1}$$

where *slerp*() is a spherical linear interpolation function, $\boldsymbol{p}_e$ is the endpoint, $\boldsymbol{R}_e$ and $\boldsymbol{t}_e$ are the overall rotation and translation parameters, and s is the interpolation factor, representing the proportion of current time to overall time.

3.1.2. Intensity calibration

LiDAR intensity values are influenced by various factors, including incidence angle, measurement distance, and surface reflectivity. To maintain stable intensity measurements, intensity calibration is necessary to eliminate the effects of these external factors:

https://doi.org/10.1017/S0263574725101926 Published online by Cambridge University Press

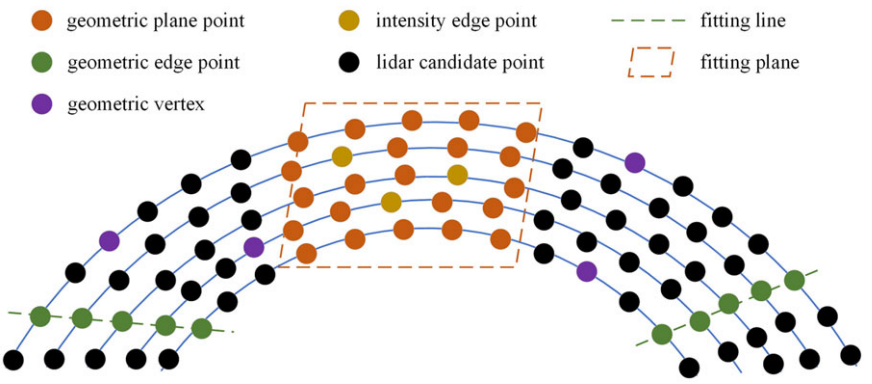


Figure 2. *Schematic diagram of feature distribution and extraction. Using the nearest neighbor search strategy to search for adjacent points of different features, and then fitting the corresponding planes and lines.*

$$I = \eta \frac{\rho \cos \alpha}{r^2} \tag{2}$$

$$\frac{I}{\cos \alpha} \propto \frac{\rho}{r^2} \tag{3}$$

where η is the constant set, ρ is the reflectivity, α represents the incidence angle, r represents distance returned from the measurement.

Therefore, the intensity measurement is primarily dependent on α, we utilize $I/\cos\alpha$ rather than I as the intensity measurement value.

3.1.3. Ground segmentation

Use dual threshold filtering method for ground point segmentation. The method first projects the point cloud onto a horizontal reference surface and partitions it into 2D grids. A filtering threshold is then determined for each grid based on the minimum height within its 3×3 neighborhood grid region. For an arbitrary point $\boldsymbol{p}_j^{(i)}$, the segmentation is formulated as:

$$\boldsymbol{p}_j^{(i)} \in \begin{cases} NG, \text{ if } h_j - h_{\min}^{(i)} > \delta_1 \text{ or } h_{\min}^{(i)} - h_{\mathrm{nmin}}^{(i)} > \delta_2 \\ G, \text{otherwise} \end{cases} \tag{4}$$

where h_j is the height of the current point, $h_{\min}^{(i)}$ and $h_{n\min}^{(i)}$ denote the minimum heights of the current grid and its neighborhood grids, δ_1 and δ_2 are two set thresholds, NG and G represent non-ground points and ground points, respectively.

3.2. Feature extraction

This section primarily focuses on extracting multiple types of geometric and intensity edge features, as illustrated in Figure 2. It begins with nearest neighbor search on the point cloud to cluster different types of points, followed by feature classification of the candidate points.

3.2.1. Geometry feature extraction

Firstly, a voxel-based method is used to downsample the segmented non-ground point clouds to reduce point density, followed by the nearest neighbor search to ensure consistency in local point cloud distribution. After these preprocessing steps, PCA is employed to extract multiple geometric features. Specifically, within a defined neighborhood radius of the current point, we search for the nearest point set and compute its covariance matrix $\sum$:

https://doi.org/10.1017/S0263574725101926 Published online by Cambridge University Press

$$\begin{aligned} \overline{\boldsymbol{u}} &= \frac{1}{|\mathrm{N}|}\sum_{i\in \mathrm{N}} \boldsymbol{p}_i \\ \sum &= \frac{1}{|\mathrm{N}|}\sum_{i\in \mathrm{N}} \left(\boldsymbol{p}_i - \overline{\boldsymbol{u}}\right)\left(\boldsymbol{p}_i - \overline{\boldsymbol{u}}\right)^T \end{aligned} \tag{5}$$

where p_i and $\overline{u}$ respectively represent the i-th point and the mean of point set N.

Using covariance matrix for eigenvalue decomposition to determine three eigenvalues and the corresponding eigenvectors:

$$\sum \boldsymbol{v}_i = \lambda_i \boldsymbol{v}_i, \quad i = 1, 2, 3 \tag{6}$$

where $\lambda_1 > \lambda_2 > \lambda_3$, $\boldsymbol{v}_1$ is the principal vector, $\boldsymbol{v}_3$ is the normal vector.

Then define local linearity σ_l, planarity σ_p and curvature σ_c as:

$$\sigma_l = \frac{\lambda_1 - \lambda_2}{\lambda_1}, \quad \sigma_p = \frac{\lambda_2 - \lambda_3}{\lambda_1}, \quad \sigma_c = \frac{\lambda_3}{\lambda_1 + \lambda_2 + \lambda_3} \tag{7}$$

Based on the magnitude of local eigenvalues and the direction of eigenvectors, three types of feature points can be extracted: planar features, linear features, and vertex features. A point is classified as a linear feature point when the local linearity within its neighborhood exceeds a predefined threshold. Similarly, points are identified as planar features when their local planarity surpasses the threshold. Afterwards, vertex features are extracted from the remaining points through curvature sampling. For ground point clouds, which contain limited vertex and linear points, only planar features are extracted. The specific classification criteria are defined as follows:

$$\boldsymbol{p}_i \in \begin{cases} P, \text{ if } \sigma_l > \sigma_{lth} \\ L, \text{ if } \sigma_p > \sigma_{pth} \\ V, \text{ if } \sigma_c > \sigma_{cth} \end{cases} \tag{8}$$

where P, L, and V, respectively, represent planar points, linear points, and vertices, σ_{lth} is the linear feature threshold, σ_{pth} is the planar feature threshold, σ_{cth} is the curvature threshold, also known as the vertex feature threshold.

3.2.2. Intensity feature extraction

Unlike three-dimensional geometric features, intensity features are one-dimensional reflectance data from LiDAR measurements. Referring to similar methods in ref. [31] and IGE-LIO [34], we extract intensity edge features by identifying points with significant intensity variations within planar point clouds through intensity difference computation, as shown in Figure 3. Although intensity planar features can represent regions of identical material composition, their extraction is susceptible to noise and viewpoint changes. Therefore, we focus exclusively on extracting more salient intensity edge features. The intensity difference between two points is formulated as:

$$\Delta I = \left| \frac{\left\|\boldsymbol{p}_i\right\|_2^2 \cdot \cos\left(\theta_j\right)}{\left\|\boldsymbol{p}_j\right\|_2^2 \cdot \cos\left(\theta_i\right)} \tilde{I}_i - \tilde{I}_j \right| \tag{9}$$

where $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$ are two candidate points, $\tilde{I}_i$ and $\tilde{I}_j$ are the corrected intensity values, and according to Eq. (3), the corrected intensity value $\tilde{I}$ is represented by $I/\cos\alpha$. θ_j and θ_j are the incident angles, and the values of cosine represent the magnitude of the point in its normal vector direction, and the formula is:

$$\cos\left(\theta_i\right) = \frac{\left(\boldsymbol{p}_i\right)^T \cdot \overline{\boldsymbol{n}}_i}{\left\|\boldsymbol{p}_i\right\|_2} \tag{10}$$

where $\overline{\boldsymbol{n}}_i$ is the normal vector, and the formula definition for $\cos\left(\theta_j\right)$ is the same.

https://doi.org/10.1017/S0263574725101926 Published online by Cambridge University Press

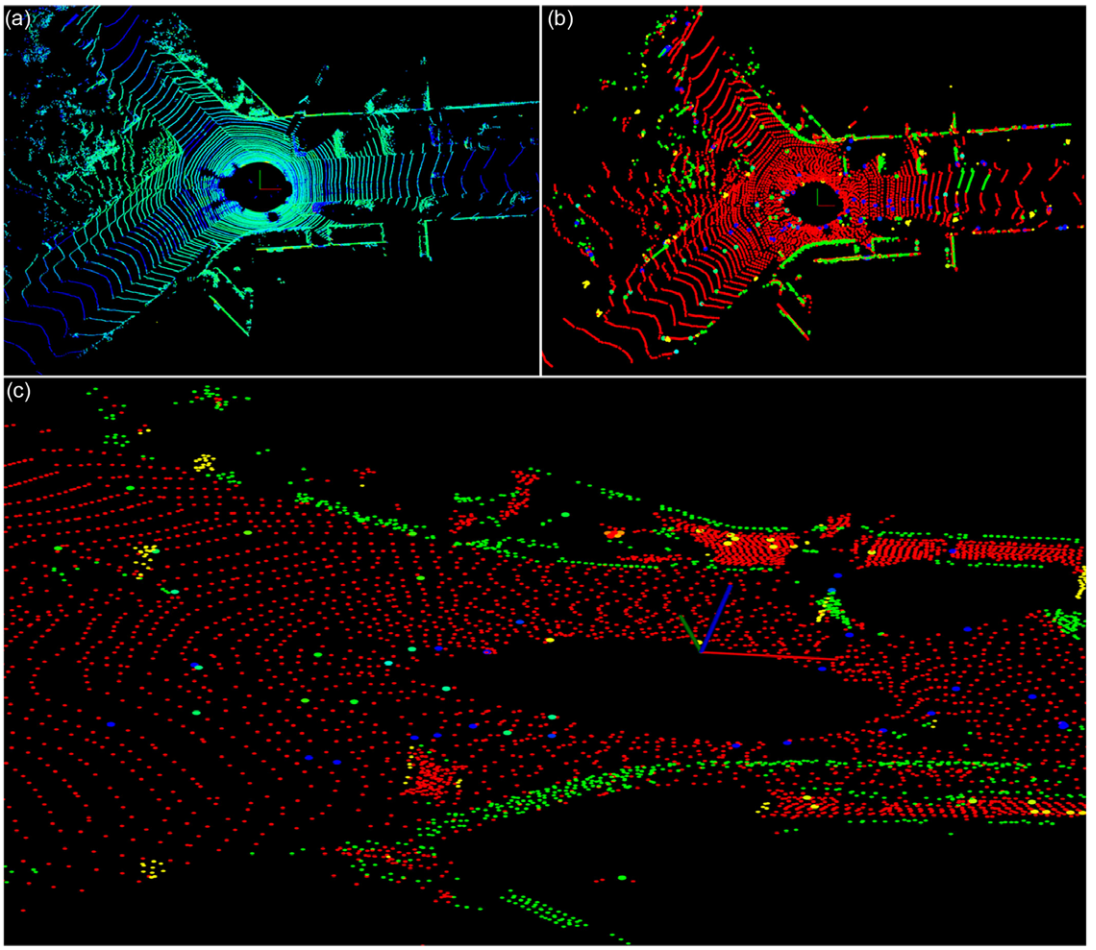


Figure 3. *Feature extraction of the current frame. (a) Original point cloud. (b) Extracted plane (red), edge (blue), vertices (purple), and intensity edge points (intensity color) feature points. (c) Enlarge the intensity edge points on the ground and display different intensity colors.*

3.2.3. Adaptive parameter

In structured environments with rich geometric information, line and planar features alone can achieve high-precision localization. However, in degenerate scenarios with sparse features, additional constraints from point features and intensity features are necessary to achieve satisfactory performance. Inspired by [26], we propose an adaptive feature extraction strategy based on the distribution and quantity of geometric and intensity features, enhancing both algorithm responsiveness and environmental robustness.

The basic principle of the adaptive parameter adjustment proposed by us for adaptive feature extraction is shown in Algorithm 1. It is mainly based on the number of geometric plane points and edge points to determine whether to extract geometric vertices and intensity edge points and adjust the threshold for feature extraction. If the number of geometric plane points and edge points exceeds the maximum threshold value, the environment is rich in geometric information. At this time, it is not necessary to extract geometric vertex and intensity edge point features, and the extraction threshold value is magnified according to the point data to extract more significant features. If it is less than the minimum threshold value, it is necessary to add geometric vertex and intensity edge point features, and reduce the relevant threshold value to extract more features. If the number of points is between the maximum and minimum, the default threshold parameter is used. The initial threshold parameters are set as follows:

https://doi.org/10.1017/S0263574725101926 Published online by Cambridge University Press

Algorithm 1. Adaptive Parameter Adjustment

Input: Geometry planar points = P, geometry linear/edge points = L, geometry planar feature threshold = σ_{pth}, geometry linear feature threshold = σ_{lth}, geometry vertex feature threshold = σ_{cth}, intensity edge feature threshold = σ_{ith}, max/min number of plane points = ($T^p_{\max}$, $T^p_{\min}$), max/min number of linear points = ($T^l_{\max}$, $T^l_{\min}$).
Output: Geometry vertices = V, intensity linear/edge points = E, adjusted threshold for geometric plane points = σ'_{pth}, adjusted threshold for geometric linear points = σ'_{lth}, adjusted threshold for geometric vertex points = σ'_{cth}, adjusted threshold for intensity edge points = σ'_{ith}.

$n_p = |P|$ // Number of geometric plane points
$n_l = |L|$ // Number of geometric linear points
diff_ratio = calculate_diff_ratio(n_p, n_l)// Calculate the difference ratio
if $n_p > T^p_{\max}$ and $n_l > T^l_{\max}$ **then**
    σ'_{pth} = increase_threshold(σ_{pth}, diff_ratio)// Increase threshold
    σ'_{lth} = increase_threshold(σ_{lth},diff_ratio)
    $\sigma'_{cth} = 0$// Do not extract vertex points
    $\sigma'_{ith} = 0$// Do not extract intensity edge points
else
    if $n_p < T^p_{\min}$ and $n_l < T^l_{\min}$ **then**
        σ'_{pth} = decrease_threshold(σ_{pth}, diff_ratio)// Decrease threshold
        σ'_{lth} = decrease_threshold(σ_{lth}, diff_ratio)
        σ'_{cth} = decrease_threshold(σ_{cth}, diff_ratio)
        σ'_{ith} = decrease_threshold(σ_{ith}, diff_ratio)
    else
        $\sigma'_{pth} = \sigma_{pth}$// Keep threshold unchanged
        $\sigma'_{lth} = \sigma_{lth}$
        $\sigma'_{cth} = \sigma_{cth}$
        $\sigma'_{ith} = \sigma_{ith}$
end if
V = extract_vertices(σ'_{cth})// Extract vertex points
E = extract_intensity_edges(σ'_{ith})// Extract intensity edge points
return $V, E, \sigma'_{pth}, \sigma'_{lth}, \sigma'_{cth}, \sigma'_{ith}$

$$\sigma_{lth} = 0.65, \sigma_{pth} = 0.65, \sigma_{cth} = 0.12, \sigma_{ith} = 0.20 \tag{11}$$

where σ_{lth}, σ_{pth} and σ_{cth} are defined by Eq. (8), and σ_{ith} is the intensity edge threshold, which is related to Eq. (9).

3.3. State estimation

For each feature point in the source frame, corresponding feature points within a defined range are identified in the target frame using nearest neighbor search, as illustrated in Figure 4. For planar and linear features, directional consistency checking of normal vectors and principal directions is performed to enhance feature association accuracy. Under multi-feature correspondence, the multi-metric approach involves estimating the optimal pose transformation through three residual metrics: point-to-point, point-to-line, and point-to-plane distance residuals. The transformation estimation $\boldsymbol{T}$ is formulated as a multi-weighted optimization problem with the following objective function:

$$\boldsymbol{T} = \underset{\boldsymbol{T}}{argmin} \left(\sum_i w_i f^2(d_i) + \sum_j w_j f^2(d_j) + \sum_k w_k f^2(d_k) \right) \tag{12}$$

https://doi.org/10.1017/S0263574725101926 Published online by Cambridge University Press

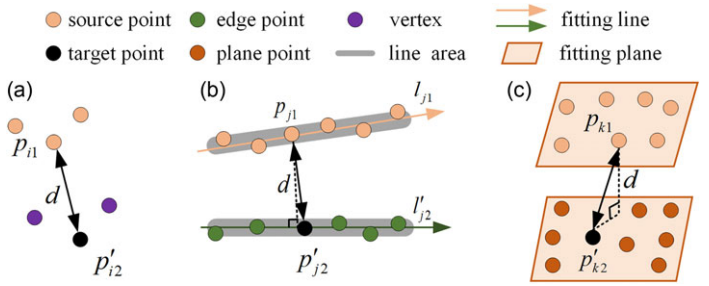


Figure 4. *Schematic illustration of three different feature distance error metrics. (a) Point to point. (b) Point to line. (c) Point to plane. Different features of the target point are represented by different colors, with superscripts representing transformed points and vectors, and black double arrows representing corresponding feature distance error vectors.*

where w_i, w_j and w_k denote the weight coefficients, d_i, d_j and d_k represent the residuals, and $f(d_i)$, $f(d_j)$ and $f(d_k)$ are the corresponding residual functions.

3.3.1. Multi-weight function

To differentiate and balance the contribution and quality of each feature type in pose estimation, we propose a novel weight function based on residuals and feature characteristics. First, a robust kernel function incorporating M-estimators [35] is employed to penalize large residuals, and a scale parameter of 1 is adopted to enhance computational efficiency. The feature weight function based on residual weights w_i^{res} is defined as:

$$\begin{aligned} w_i^{\text{res}} &= \varepsilon \left(\varepsilon^2 + 1\right)^{-\frac{1}{2}} \\ \varepsilon &= \frac{f}{\sigma} \end{aligned} \tag{13}$$

where f is the residual in Eq. (11), ε is the normalized residual, and σ is the threshold of noise.

For geometric planar features, both intensity values and smoothness contribute to feature quality. Therefore, the weight for geometric planar features can be expressed as:

$$\begin{aligned} lw_i^{GPP} &= \frac{2.0}{2.0 - e^{-2\sigma_I c_i}} \\ c_i &= \frac{1}{|N| \cdot \|\mathbf{p}_i\|} \left\| \sum_{p_j \in M, i \neq j} \left(\mathbf{p}_i - \mathbf{p}_j\right) \right\| \end{aligned} \tag{14}$$

where c_i is the smoothness of the point $\mathbf{p}_i$, and $\mathbf{p}_j$ is the neighborhood point of $\mathbf{p}_i$, M is the set of points around $\mathbf{p}_j$, and N is the number of points in the M set, and σ_I is the parameter of the Gaussian distribution of intensity values, which is described as:

$$\begin{aligned} \mu &= \frac{1}{N} \sum_{j=1}^{N} I_j \\ \sigma_I^2 &= \frac{1}{N-1} \sum_{j=1}^{N} \left(I_j - \mu\right)^2 \end{aligned} \tag{15}$$

where I_j represents the intensity value of point j, μ is the average intensity of the point set, and σ_I^2 is its variance.

https://doi.org/10.1017/S0263574725101926 Published online by Cambridge University Press

For geometric edge features, smoothness has a more significant impact; therefore, the weight is defined as:

$$w_i^{GEP} = \frac{2.0}{1.0 - e^{-c_i}} \tag{16}$$

Similarly, the weight for geometric vertex features is defined as:

$$w_i^{GVP} = \frac{2.0}{1.0 + e^{-c_i}} \tag{17}$$

The weight of intensity variations between vector points is defined as:

$$w_i^{IEP} = \frac{2.0}{1.0 + e^{-\Delta I/255}} \tag{18}$$

where ΔI denotes the intensity difference.

Therefore, the final weight is:

$$w_i = w_i^{res} \times w_f \tag{19}$$

where $w_f = w_i^{GPP} \cdot w_i^{GEP} \cdot w_i^{GVP} \cdot w_i^{IEP}$ is the weight of various features. Similarly, w_j and w_k.

The larger the weight value of a feature point, the more representative it is, thereby improving the results of pose estimation.

3.3.2. Multi-metric estimation

Inspired by MULLS [23] and [26], the calculation formula for point-to-point, point-to-line, and point-to-plane error functions is:

$$\begin{aligned} f(d_i) &= \left\| \boldsymbol{p}_{i1} - \boldsymbol{T}\boldsymbol{p}'_{i2} \right\|_{\sigma} \\ f(d_j) &= \left\| \left(\boldsymbol{p}_{j1} - \boldsymbol{T}\boldsymbol{p}'_{j2}\right) \times \boldsymbol{l}_c \right\|_2 \\ f(d_k) &= \left| \left(\boldsymbol{p}_{k1} - \boldsymbol{T}\boldsymbol{p}'_{k2}\right) \times \boldsymbol{n}_c \right| \end{aligned} \tag{20}$$

where $\sigma = (\sum_{i1} + \boldsymbol{T}\sum'_{i2}\boldsymbol{T}^T)^{-1}$, $\boldsymbol{p}$ is the corresponding feature point, $\boldsymbol{l}_c$, $\boldsymbol{n}_c$ and $\sum_i$ represent the direction vector, normal vector, and covariance matrix, respectively.

The final pose transformation $\boldsymbol{T}$ is estimated by minimizing geometric and intensity residuals:

$$\boldsymbol{T} = \underset{T}{argmin} \left(\begin{array}{l} \sum_i w_i \left(\left\| \boldsymbol{p}_{i1} - \boldsymbol{T}\boldsymbol{p}'_{i2} \right\|_{\sigma} \right)^2 + \sum_j w_j \left(\left\| \left(\boldsymbol{p}_{j1} - \boldsymbol{T}\boldsymbol{p}'_{j2}\right) \times \boldsymbol{l}_c \right\|_2 \right)^2 + \\ \sum_k w_k \left(\left| \left(\boldsymbol{p}_{k1} - \boldsymbol{T}\boldsymbol{p}'_{k2}\right) \times \boldsymbol{n}_c \right| \right)^2 \end{array} \right) \tag{21}$$

3.3.3. Two-step decoupling

Inspired by LeGO-LOAM [21], the final pose transformation can be resolved through the two-step decoupled Levenberg-Marquardt (LM) method. As shown in Figure 5, the estimation of [Z, Roll, Pitch] is first performed using plane feature constraints, followed by the estimation of [X, Y, Yaw] using edge and vertex feature constraints. Finally, these are combined to obtain the six degrees of freedom pose estimation [X, Y, Z, Roll, Pitch, Yaw].

3.4. Global optimization

Use TEASER [36] based loop descriptors for loop detection. After completing loop detection and consistency verification, if there is a loop closure, perform pose graph optimization, correct the accumulated drift of the odometer, and finally output a dense global map.

https://doi.org/10.1017/S0263574725101926 Published online by Cambridge University Press

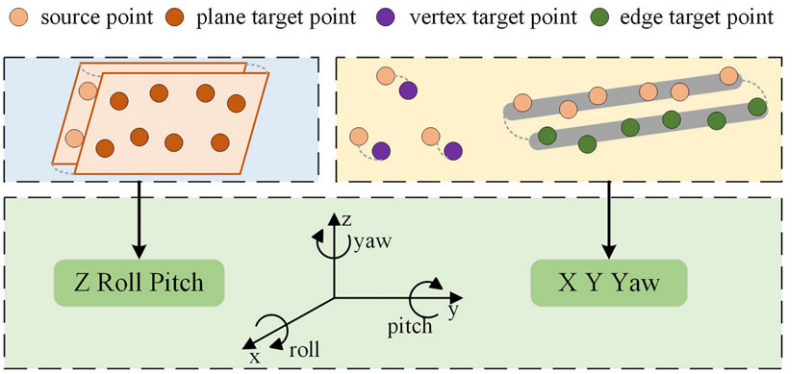


Figure 5. *Schematic diagram of two-step decoupling for pose estimation. Plane feature point matching is used to estimate three pose variables, while vertex and edge feature point matching is used to estimate the other three pose variables.*

4. Experiments and results

In this section, we conducted a series of qualitative and quantitative experiments on two public datasets to evaluate the proposed method, comprising unstructured environments that pose challenges for traditional LiDAR SLAM algorithms to manage, such as open highways, city streets, and so on.

4.1. Experimental setup

To ensure fairness and assess the computational cost of various methods, all algorithms in the public dataset were tested on desktop computers equipped with Intel i9-9900X 10-core CPU and 46 GB RAM. We employ two widely used SLAM error assessment metrics: Absolute Trajectory Error (ATE) and Relative Pose Error (RPE). Among them, RPE can be further divided into Relative Translation Error (RTE) and Relative Rotation Error (RRE). All the above indicators use their root mean square error (RMSE) data, and the relevant calculation formula [37] is as follows:

$$\begin{aligned}
&\mathrm{ATE} = \boldsymbol{Q}_i^{-1}\boldsymbol{S}\boldsymbol{P}_i \\
&\mathrm{RPE} = \left(\boldsymbol{Q}_i^{-1}\boldsymbol{Q}_{i+\Delta}\right)^{-1}\left(\boldsymbol{P}_i^{-1}\boldsymbol{P}_{i+\Delta}\right) \\
&\mathrm{RMSE}\,(ATE) = \left(\frac{1}{m}\sum_{i=1}^{m}\|trans\,(ATE)\|^2\right)^{1/2} \\
&\mathrm{RMSE}\,(ARE) = \left(\frac{1}{m}\sum_{i=1}^{m}\|rot\,(ATE)\|^2\right)^{1/2} \\
&\mathrm{RMSE}\,(RTE) = \left(\frac{1}{m}\sum_{i=1}^{m}\|trans\,(RPE)\|^2\right)^{1/2} \\
&\mathrm{RMSE}\,(RRE) = \left(\frac{1}{m}\sum_{i=1}^{m}\|rot\,(RPE)\|^2\right)^{1/2}
\end{aligned} \tag{22}$$

where $\boldsymbol{P}_i$ and $\boldsymbol{Q}_i$ are the pose to be estimated and the pose truth, $\boldsymbol{S}$ is the transformation matrix, i represents time, and Δ represents the fixed time difference of the interval.

4.2. Public datasets

KITTI The KITTI dataset [38] is a publicly available dataset jointly created by the Karlsruhe Institute of Technology in Germany and the Toyota Technical Research Institute in the United States. It collects actual traffic scene data through fully equipped vehicles, as shown in Figure 6. It is a commonly used

https://doi.org/10.1017/S0263574725101926 Published online by Cambridge University Press



Figure 6. *The data collection platform for the KITTI dataset. Including devices such as 3D LiDAR, cameras, IMU, and GPS, it can provide trajectory truth values for multiple sequences.*



Figure 7. *The data collection platform for the MulRan dataset. Equipped with a single radar and 3D LiDAR sensor, it can provide multi-mode distance measurement and trajectory truth.*

publicly available dataset for evaluating SLAM algorithms. HDL-64E LiDAR is utilized for data collection from 22 sequences in various scenarios, including urban, rural, and highway scenes, and provides 11 sequences with ground truth (00-10).

MulRan The MulRan dataset [39] is a multimodal dataset specifically designed for urban environments, collected using Radar and LiDAR sensors, as shown in Figure 7. This dataset collects four different environments, each of which has three subsequences. It also contains multi-session sequences of repeated tracks in the urban scene, and provides a six-dimensional real ground truth trajectory.

https://doi.org/10.1017/S0263574725101926 Published online by Cambridge University Press

Table I. *RMSE of ATE (m) comparison of pure geometric algorithms on KITTI dataset.*

Method	00	02	03	05	07	09
ALOAM	8.392	97.975	0.855	3.281	0.814	1.649
LeGO-LOAM	4.098	86.677	1.073	1.737	0.937	2.440
MULLS	1.780	7.140	0.681	0.815	0.573	2.087
Proposed	**1.265**	**5.572**	**0.616**	**0.695**	**0.326**	**1.299**

Table II. *RMSE of RTE (m) and RRE (deg) comparison of pure geometric algorithms on KITTI dataset.*

	ALOAM		LeGO-LOAM		MULLS		Proposed	
Sequence	RTE (m)	RRE (deg)	RTE (m)	RRE (deg)	RTE (m)	RRE (deg)	RTE (m)	RRE (deg)
00	1.217	1.781	0.368	2.332	0.039	0.783	**0.030**	**0.132**
02	1.537	1.426	0.647	2.010	0.046	0.447	**0.033**	**0.100**
05	1.197	1.254	0.102	1.761	0.063	0.082	**0.017**	**0.068**
06	1.631	1.522	0.327	2.096	0.082	0.054	**0.015**	**0.042**
08	1.184	1.445	0.328	1.444	0.063	0.093	**0.040**	**0.080**
10	1.177	1.296	0.259	1.827	0.069	0.099	**0.021**	**0.096**

4.3. Evaluation on public datasets

Quantitative experiments are conducted on both datasets, comparing our algorithm with basic and state-of-the-art pure geometric algorithms as well as intensity-assisted algorithms, including ALOAM,[1] LeGO-LOAM [21], MULLS [23], SC-LeGO-LOAM,[2] Intensity-SLAM [30], and ISC-LOAM [40].

4.3.1. Comparison with pure geometric algorithms

On the KITTI dataset, select sequences 00, 02, 03, 05, 07, and 09 are selected for the comparison of ATE, while select sequences 00, 02, 05, 06, 08, and 10 are chosen for the comparison of RTE and RRE, along with the corresponding trajectory plots. The comparisons are made with the pure geometric algorithms, and the results are presented in Table I, Table II, and Figure 8. The results indicate that in most sequences in the KITTI dataset, our proposed algorithm outperforms these three pure geometry algorithms in terms of accuracy. Specifically, in Table I, on the complex and comprehensive sequence 00, our odometry achieves an APE accuracy of 1.265 m, which is an improvement of 84.93% over ALOAM's 8.392 m, 69.13% over LeGO-LOAM's 4.098 m, and 28.93% over MULLS' 1.780 m. Our method also shows enhanced accuracy compared to the other methods in the remaining sequences. In Table II, the relative pose accuracy shows that our proposed algorithm significantly outperforms the comparison algorithms. In the trajectory plot shown in Figure 8, our algorithm is closer to the ground truth, particularly in sparse scenarios such as turns and roads, indicating that our proposed algorithm has higher accuracy and robustness.

On the MulRan dataset, representative sequences DCC01, KAIST01, and Riverside01 are selected for algorithm comparison experiments. As shown in Table III, our algorithm outperforms both SC-LeGO-LOAM and MULLS in most of the two types of accuracy metrics, particularly in RTE, which demonstrates the effectiveness of our adaptive feature extraction and two-step decoupling.

4.3.2. Comparison with intensity-assisted algorithms

On the other hand, we select sequences 01, 04, 05, 07, 08, and 09 from the KITTI dataset to compare the ATE and RTE accuracy with state-of-the-art intensity-assisted algorithms, and the results are shown in

[1] https://github.com/HKUST-Aerial-Robotics/A-LOAM.
[2] https://github.com/irapkaist/SC-LeGO-LOAM.

https://doi.org/10.1017/S0263574725101926 Published online by Cambridge University Press

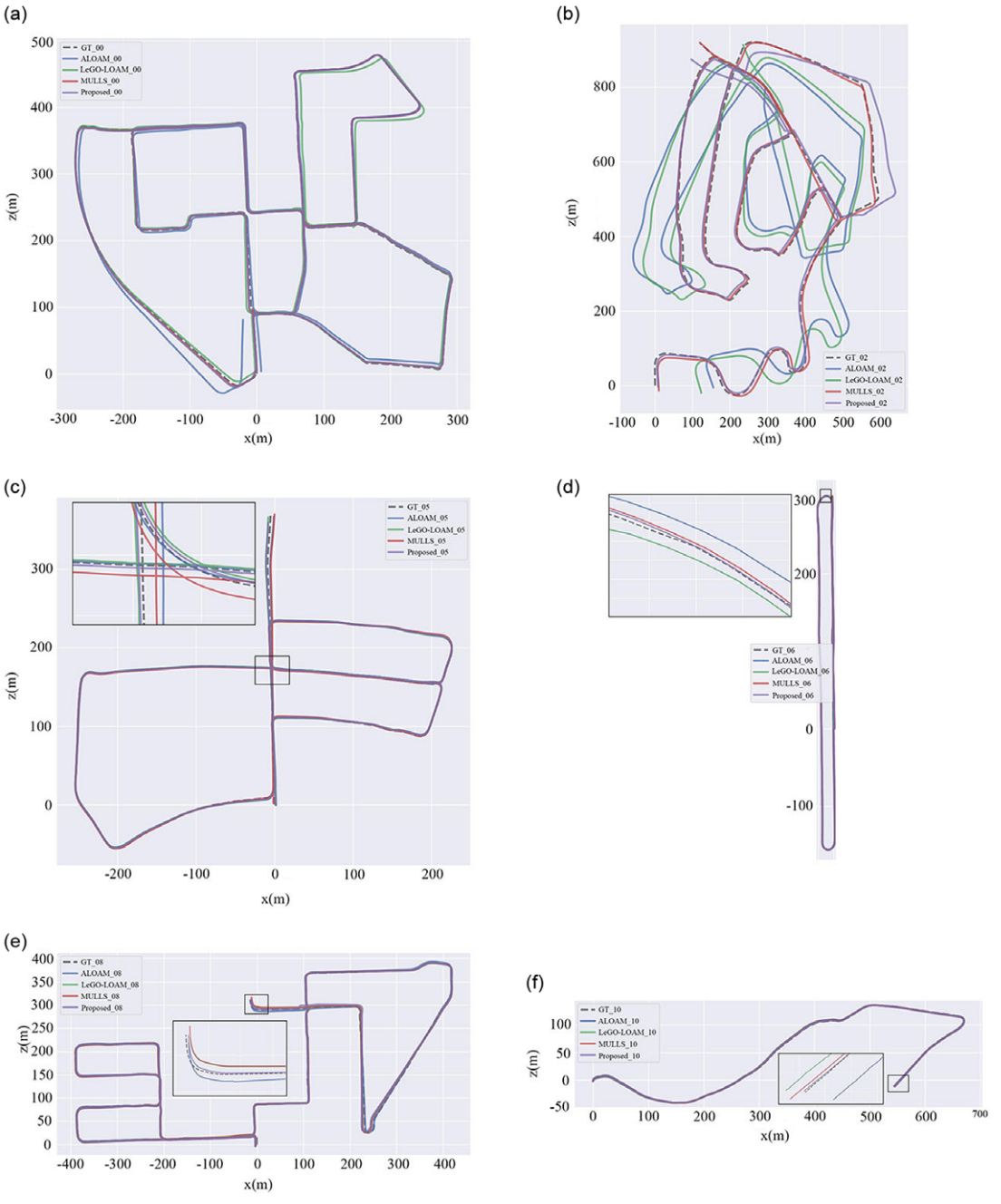


Figure 8. *The estimated trajectory is compared with ALOAM, LeGO-LOAM, and MULLS on some KITTI dataset sequences, and their respective ground truths are also provided. (a) Seq 00. (b) Seq 02. (c) Seq 05. (d) Seq 06. (d) Seq 08. (f) Seq 10.*

Table IV. As shown in the table, our algorithm has demonstrated significantly superior accuracy in most sequences, especially with a more significant improvement in RTE accuracy, indicating the effectiveness of our algorithm. On the 04 sequence, our algorithm's ATE is lower than these two types of algorithms because the sequence is a short linear path with a shorter running time. These three algorithms have

https://doi.org/10.1017/S0263574725101926 Published online by Cambridge University Press

Table III. *RMSE of ATE (m) and RTE(m) comparison of geometric algorithms on MulRan dataset.*

	SC-LeGO-LOAM		MULLS		Proposed	
Sequence	**ATE (m)**	**RTE (m)**	**ATE (m)**	**RTE (m)**	**ATE (m)**	**RTE (m)**
DCC01	**16.118**	1.381	17.852	1.873	16.522	**1.115**
KAIST01	28.036	1.201	26.634	1.593	**19.697**	**1.092**
Riverside01	69.620	1.859	62.473	2.550	**54.159**	**1.541**

Table IV. *RMSE of ATE (m) and RTE(m) comparison of intensity-assisted algorithms on KITTI dataset.*

Sequence	**Intensity-SLAM**		**ISC-LOAM**		**Proposed**	
	ATE (m)	**RTE (m)**	**ATE (m)**	**RTE (m)**	**ATE (m)**	**RTE (m)**
00	20.582	3.217	19.059	3.217	**2.021**	**0.046**
04	0.357	2.055	**0.342**	2.053	0.365	**0.026**
05	2.398	1.198	1.801	1.200	**0.695**	**0.017**
07	0.716	0.999	0.635	1.004	**0.326**	**0.015**
08	4.727	1.183	4.468	1.183	**2.543**	**0.040**
09	2.086	1.549	1.688	1.550	**1.299**	**0.088**

similar ATE accuracy on this sequence, so RTE is more suitable for evaluating this sequence. Our algorithm has a more obvious high accuracy on RTE indicating that our algorithm still has superiority on this sequence.

4.3.3. Unstructured scenes mapping

To illustrate the mapping performance of the algorithm we proposed in unstructured scenes with sparse features, we conducted qualitative and quantitative experiments on sequences 01 and 10 of the KITTI dataset containing unstructured features, as shown in Figure 9. It can be seen that even in challenging highway scenarios and small-town roads, our algorithm can establish accurate point maps with low drift poses.

4.4. Ablation study

To analyze the performance of our proposed adaptive feature extraction method, we performed ablation experiments on odometer accuracy. Perform the localization accuracy experiment using the KITTI-05 sequence and compare it with pure geometry method, intensity odometry estimation method, and MULLS algorithm. There is no loop detection function used here, as shown in Table V. Although our proposed method increases some time consumption, the ATE error is 0.108% and the ARE error is 1.195 rad, which are better than other algorithms, indicating that our method is effective.

On the other hand, adaptive parameters and two-step decoupling are the focus of an ablation study conducted using the KITTI-07 sequence for experimentation, where loop closure detection functionality is used. For convenience, we use AFE to represent adaptive feature extraction and TSD to represent two-step decoupling. As shown in Table VI, Proposed[1] does not use AFE or TSD, Proposed[2] uses AFE but not TSD, and Proposed[3] uses TSD but not AFE. From the results in the table, it can be seen that introducing these two methods significantly enhances both time efficiency and accuracy, particularly the incorporation of two-step decoupling, which greatly improves the accuracy of RTE.

https://doi.org/10.1017/S0263574725101926 Published online by Cambridge University Press

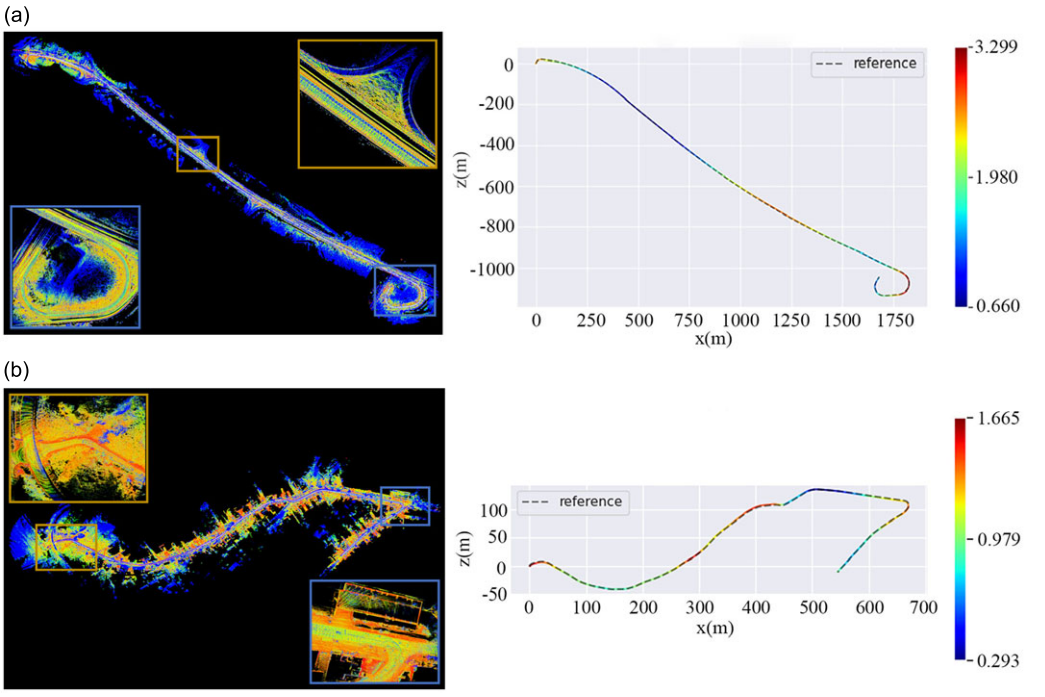


Figure 9. *Mapping and trajectory results of typical unstructured scenes. (a) KITTI-01 sequence is a highway scenario. (b) KITTI-10 sequence is an urban scenario.*

4.5. Runtime analysis

The detailed time consumption of the algorithm is presented in Table VII and Figure 10. In the KITTI-00 sequence, the feature extraction module has an average processing time of 33.11 ms per frame, the matching and registration module takes an average of 18.03 ms per frame, the map update module averages 2.71 ms per frame, and the loop closure detection module averages 9.05 ms per frame. Overall, our algorithm takes an average of 62.9 ms per frame, which meets the requirement for a LiDAR operating frequency of 10 Hz.

4.6. Validation on practical robot

To assess the effectiveness of our proposed algorithm in actual unstructured environments, we performed qualitative tests using the Jueying Lite2 quadruped robot in the university park, as shown in Figure 11. The robot has a 3D LiDAR with VLP-16, a D435i depth camera, and an NVIDIA Jetson Xavier NX GPU processor, which effectively processes the measurement data.

Figure 12(a) shows a segment of the actual scene captured on Gaode Map. Figure 12(b) displays the final dense mapping result, with the robot's trajectory represented by red dots. The overall results are consistent with the actual scene, and the trajectory is very continuous, showing no significant drift.

Figures 12(c) and (d) present enlarged local mapping results, where the object contours in the local mapping are also quite clear, without blurriness or ghosting. Therefore, the proposed method can accurately localize and construct dense maps in such unstructured scenarios.

5. Conclusion

In this paper, we propose a novel multi-metric LiDAR SLAM system that effectively addresses challenges in unstructured environments while achieving high localization accuracy. By adaptively extracting various types of geometric features and prominent intensity edge features, we enhance the

https://doi.org/10.1017/S0263574725101926 Published online by Cambridge University Press

Table V. *The time and accuracy of odometer experiments.*

Method	Time (ms)	ATE (%)	ARE (rad)
Odometry(only-geometric)	63.01	1.108	1.198
Odometry(with-intensity)	74.14	0.139	1.200
MULLS	**52.03**	0.127	2.094
Proposed	66.67	**0.108**	**1.195**

Table VI. *The time and accuracy of ablation experiments.*

Method	AFE	TSD	Time (ms)	ATE (%)	RTE (m)
Proposed[1]	✗	✗	73.89	0.087	0.129
Proposed[2]	✔	✗	58.61	0.064	0.098
Proposed[3]	✗	✔	69.75	0.051	0.070
Proposed	✔	✔	**52.84**	**0.047**	**0.015**

Table VII. *Time consumption per frame for each module.*

Method	Time (ms)
Feature extraction	33.11
Registration	18.03
Map updating	2.71
Loop closure	9.05

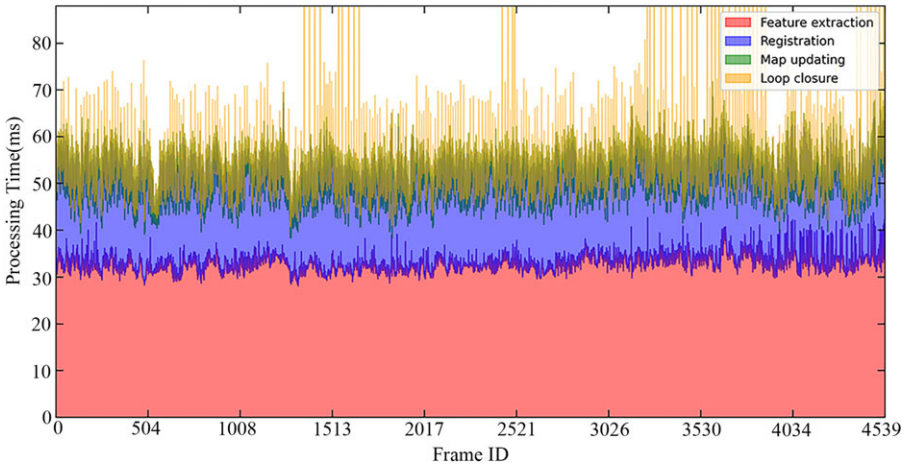


Figure 10. *The running time per frame of each module in the proposed algorithm.*

capability to capture environmental information. We then perform multi-metric error calculations for point, line, and plane features, and design a new multi-weight function while employing a two-step decoupling strategy for pose variables to improve localization accuracy. Experimental results on various datasets with unstructured scenes demonstrate that our proposed algorithm achieves superior accuracy and real-time performance compared to state-of-the-art pure geometric and intensity-assisted algorithms. Compared with the benchmark algorithm, our proposed algorithm has improved ATE accuracy by 28.93% and real-time performance can reach 62.9 ms per frame. In future work, we intend

https://doi.org/10.1017/S0263574725101926 Published online by Cambridge University Press



Figure 11. *The Jueying Lite2 quadruped platform. It is equipped with a VLP-16 LiDAR, a D435i camera, and an NVIDIA Xavier NX processor.*

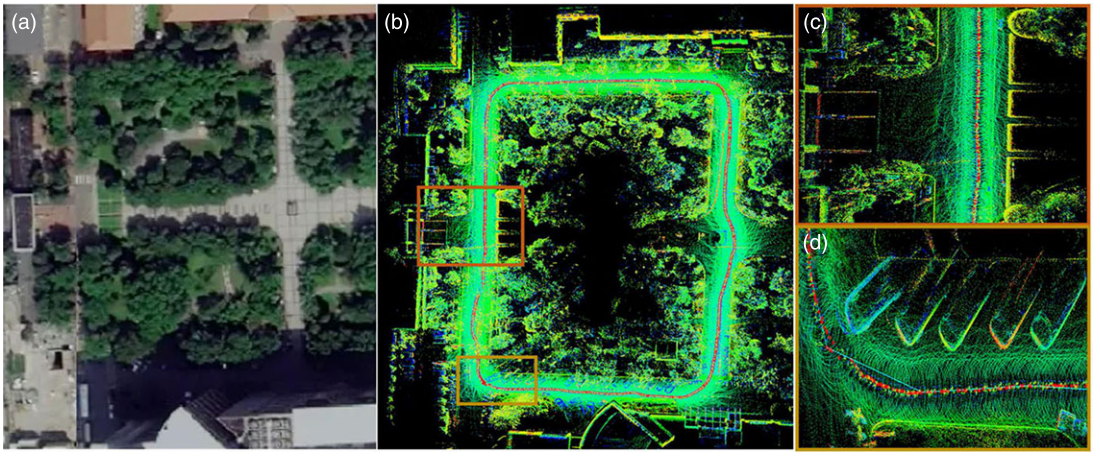


Figure 12. *Mapping in real unstructured campus scenes. (a) Aerial view of a real scene. (b) Algorithm mapping results. (c) and (d) are enlarged partial views.*

to utilize other sensors for data fusion to further enhance the adaptability of our algorithm in intricate environments.

Author contributions. All authors participated in the implementation of this study. The initial version of the manuscript was written by Xu Zhenghui, with Li Jian and Tang Ling providing assistance with experimental design, Wei Shimin and Chen Huanlong providing equipment, and all authors providing feedback on the early drafts of the manuscript. All authors have reviewed and approved the final manuscript.

Financial support. This work was supported by the funding from the Shanghai Institute of Aerospace Systems Engineering (No. SAST2022-015).

Competing interests. The authors declare that they have no conflicts of interest to disclose.

Ethical approval. None.

https://doi.org/10.1017/S0263574725101926 Published online by Cambridge University Press

References

[1] Y. Wang, Y. Tian, J. Chen, K. Xu and X. Ding, "A survey of visual SLAM in dynamic environment: The evolution from geometric to semantic approaches," *IEEE Trans. Instrum. Meas.* **73**, 1–21 (2024).

[2] C. Da, Z. Chen, T. Song and Y. Lu, "Tightly coupled SLAM system for indoor complex scenes," *Robotica*, 1–14 (2025).

[3] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. Robot.* **32**(6), 1309–1332 (2016).

[4] D. Cai, R. Li, Z. Hu, J. Lu, S. Li and Y. Zhao, "A comprehensive overview of core modules in visual SLAM framework," *Neurocomputing* **590**, 127760 (2024).

[5] Y. Liu, C. Wang, H. Wu, Y. Wei, M. Ren, C. Zhao, "Improved LiDAR localization method for mobile robots based on multi-sensing," *Remote Sens. BASEL* **14**(23), 6133 (2022).

[6] H. Pu, J. Luo, G. Wang, T. Huang, H. Liu and J. Luo, "Visual SLAM integration with semantic segmentation and deep learning: A review," *IEEE Sens. J.* **23**(19), 22119–22138 (2023).

[7] C. Debeunne and D. Vivet, "A review of visual-LiDAR fusion based simultaneous localization and mapping," *Sensors* **20**(7), 2068 (2020).

[8] Y. Zhang, P. Shi and J. Li, "3D LiDAR SLAM: A survey," *Photogrammetr. Record* **39**(186), 457–517 (2024).

[9] D. T. Fasiolo, L. Scalera and E. Maset, "Comparing LiDAR and IMU-based SLAM approaches for 3D robotic mapping," *Robotica* **41**(9), 2588–2604 (2023).

[10] X. Huang, G. Mei, J. Zhang, R. Abbas, "A comprehensive survey on point cloud registration, arXiv preprint arXiv: 2103.02690 (2021).

[11] J. Zhang, Y. Yao and B. Deng, "Fast and robust iterative closest point," *IEEE Trans. Pattern Anal. Mach. Intell.* **44**(7), 3450–3466 (2021).

[12] S. Liu, E. Sun and X. Dong, "SLAMB&MAI: A comprehensive methodology for SLAM benchmark and map accuracy improvement," *Robotica* **42**(4), 1039–1054 (2024).

[13] J. Behley and C. Stachniss, "Efficient surfel-based SLAM using 3D laser range data in urban environments," **In:** *Robotics: science and systems* (2018) p. 59.

[14] D. Rozenberszki and A.L. Majdik. "LOL: Lidar-only Odometry and Localization in 3D Point Cloud Maps." **In:** *2020 IEEE International Conference on Robotics and Automation (ICRA). IEEE* (2020).

[15] P. Dellenbach, J.E. Deschaud, M. Paris, B. Jacquet, F. Goulette. "CT-ICP: Real-time Elastic LiDAR Odometry with Loop Closure." **In:** *2022 International Conference on Robotics and Automation (ICRA). IEEE* (2022).

[16] P. Biber and W. Straßer, "The Normal Distributions Transform: A New Approach to Laser Scan Matching," **In:** *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No. 03CH37453)* (IEEE, 2003).

[17] 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.* **16**(2), 1729881419841532 (2019).

[18] S. Bouraine, A. Bougouffa and O. Azouaoui. "NDT-PSO, a New NDT based SLAM Approach Using Particle Swarm Optimization." **In:** *2020 16th International Conference on Control, Automation, Robotics and Vision (ICARCV). IEEE* (2020).

[19] Z. Zhou, C. Zhao, D. Adolfsson, S. Su, Y. Gao, T. Duckett, L. Sun. "NDT-transformer: Large-scale 3D Point Cloud Localisation Using the Normal Distribution Transform Representation." **In:** *2021 IEEE International Conference on Robotics and Automation (ICRA). IEEE* (2021).

[20] J. Zhang and S. Singh, "LOAM: Lidar odometry and mapping in real-time," *Robot. Sci. Syst*. **2**(9), 1–9 (2014).

[21] T. Shan and B. Englot. "Lego-loam: Lightweight and Ground-optimized LiDAR Odometry and Mapping on Variable Terrain." **In:** *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE* (2018).

[22] H. Wang, C. Wang, C. Chen, L. Xie. "F-loam: Fast LiDAR Odometry and Mapping." **In:** *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE* (2021).

[23] Y. Pan, P. Xiao, Y. He, Z. Shao, Z. Li. "MULLS: Versatile LiDAR SLAM via Multi-metric Linear Least Square." **In:** *2021 IEEE International Conference on Robotics and Automation (ICRA). IEEE* (2021).

[24] M. Greenacre, P. J. F. Groenen, T. Hastie, A. I. D'Enza, A. Markos and E. Tuzhilina, "Principal component analysis," *Nat. Rev. Methods Primers* **2**(1), 100 (2022).

[25] S. Chen, H. Ma, C. Jiang, B. Zhou, W. Xue, Z. Xiao and Q. Li, "NDT-LOAM: A real-time LiDAR odometry and mapping with weighted NDT and LFA," *IEEE Sens. J.* **22**(4), 3660–3671 (2021).

[26] M. Xu, S. Lin, J. Wang and Z. Chen, "A LiDAR SLAM system with geometry feature group based stable feature selection and three-stage loop closure optimization," *IEEE Trans. Instrum. Meas.* **72**, 1–10 (2023).

[27] L.I. Shuaixin, L.I. Jiuren, T. Bin, C. Long, W. Li, L.I. Guangyun, "A laser SLAM method for unmanned vehicles in point cloud degenerated tunnel environments," *Acta Geodaetica Cartogr. Sin.* **50**(11), 1487 (2021).

[28] S. Sato, Y. Yao, T. Yoshida, T. Kaneko, S. Ando, J. Shimamura. "Unsupervised Intrinsic Image Decomposition with LiDAR Intensity." **In:** *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2023).

[29] Y. S. Park, H. Jang and A. Kim. "I-LOAM: Intensity Enhanced LiDAR Odometry and Mapping." **In:** *2020 17th International Conference on Ubiquitous Robots (UR). IEEE* (2020).

[30] H. Wang, C. Wang and L. Xie, "Intensity-slam: Intensity assisted localization and mapping for large scale environment," *IEEE Robot. Autom. Lett.* **6**(2), 1715–1721 (2021).

[31] J. Li, R. Ji, X. Liang, S. S. Ge and H. Yan, "An intensity-augmented LiDAR-inertial SLAM for solid-state LiDARs in degenerated environments," *IEEE Trans. Instrum. Meas.* **71**, 1–10 (2022).

https://doi.org/10.1017/S0263574725101926 Published online by Cambridge University Press

[32] Z. Dai, J. Zhou, T. Li, H. Yao, S. Sun and X. Zhu, "An intensity-enhanced LiDAR SLAM for unstructured environments," *Meas. Sci. Technol.* **34**(12), 125120 (2023).
[33] X. Zhang, H. Zhang, C. Qian, B. Li and Y. Cao, "A LiDAR-intensity SLAM and loop closure detection method using an intensity cylindrical-projection shape context descriptor," *Int. J. Appl. Earth Observ. Geoinf.* **122**, 103419 (2023).
[34] Z. Chen, H. Zhu, B. Yu, C. Jiang, C. Hua, X. Fu and X. Kuang, "IGE-LIO: Intensity gradient enhanced tightly-coupled LiDAR-inertial odometry," *IEEE Trans. Instrum. Meas.* **73**, 1–11 (2024).
[35] J. Guo, P. V. K. Borges, C. Park and A. Gawel, "Local descriptor for robust place recognition using lidar intensity," *IEEE Robot. Autom. Lett.* **4**(2), 1470–1477 (2019).
[36] H. Yang, J. Shi and L. Carlone, "Teaser: Fast and certifiable point cloud registration," *IEEE Trans. Robot.* **37**(2), 314–333 (2020).
[37] W. Lu, Y. Zhou, G. Wan, S. Hou, S. Song. "L3-net: Towards Learning based LiDAR Localization for Autonomous Driving." **In:** *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019).
[38] A. Geiger, P. Lenz and R. Urtasun. "Are We Ready for Autonomous Driving? The Kitti Vision Benchmark Suite." **In:** *2012 IEEE Conference on Computer Vision and Pattern Recognition. IEEE* (2012).
[39] G. Kim, Y.S. Park, Y.H. Cho, J. Jeong, A. Kim. "Mulran: Multimodal Range Dataset for Urban Place Recognition." **In:** *2020 IEEE International Conference on Robotics and Automation (ICRA). IEEE* (2020).
[40] H. Wang, C. Wang and L. Xie. "Intensity Scan Context: Coding Intensity and Geometry Relations for Loop Closure Detection." **In:** *2020 IEEE International Conference on Robotics and Automation (ICRA). IEEE* (2020).

Cite this article: Z. Xu, J. Li, S. Wei, L. Tang and H. Chen (2025). "A multi-metric LiDAR Simultaneous Localization and Mapping system based on intensity feature assistance and two-step decoupling estimation", Robotica **43**, 2872–2891. https://doi.org/10.1017/S0263574725101926

https://doi.org/10.1017/S0263574725101926 Published online by Cambridge University Press