E-LOAM: LiDAR Odometry and Mapping With Expanded Local Structural Information

Hongliang Guo, Jiankang Zhu, and Yunping Chen, *Member, IEEE*

***Abstract*—This paper investigates the real time LiDAR odometry and mapping (LOAM) problem in unstructured environments. We propose E-LOAM (LOAM with Expanded Local Structural Information), a paradigm which expands the pre-extracted geometric information with local point cloud information around the geometric feature points. State-of-the-art approaches usually extract *pointed* geometric features as the only correspondence primitives for point cloud scan-to-scan and scan-to-map registration. We argue that, in unstructured environments, sometimes, the extracted geometric features are too sparse for adequate point cloud registration. Therefore, E-LOAM expands the 'pointed' geometric correspondence primitives with the point clouds around them, i.e., the local point clouds in the voxel around the feature point. The local point clouds, approximated by a multivariate normal distribution, offer additional local *structural* information, on top of the *pointed* geometric information. Additionally, to enrich the sparse geometric features, we make use of the intensity information of point clouds, and extract the places with high intensity variations as additional feature points. Experimental results with the KITTI dataset show the efficacy of E-LOAM, when compared with state of the arts. We further implement E-LOAM on a real robot platform, and evaluate E-LOAM with in-field tests.**

***Index Terms*—E-LOAM, expanded local structural information, scan-to-scan registration, point cloud registration, correspondence primitive.**

I. Introduction

AUTONOMOUS robots have been seemingly integrated into human lives for various kinds of applications, ranging from deliveries, households, to surveillance and farming. Out of all application scenarios, one essentially required functionality is to localize the robot within unknown environments. The easiest way of localizing a robot is through accurate GPS signals to get the global coordinate information. However, one cannot always expect the robot to have available GPS signals during navigation, e.g., the indoor navigation task usually lacks the GPS signal. Therefore, it is appealing for the robot to be able to construct a map in the target environment and localize itself within the self-constructed map. The prevailing solution for self-map construction and localization within an unknown environment is through Simultaneous Localization And Mapping (SLAM). With SLAM, the robot navigates within the environment, constructs a map, and in the meanwhile, localizes itself within the self-constructed map.

One of the most commonly used sensors for SLAM implementation is Light Detection And Ranging (LiDAR), in that LiDAR directly provides range information which is essential in the mapping and localization process, and it is more robust to light changes when compared with vision sensors, i.e., cameras. Moreover, the price of a LiDAR sensor has been decreasing constantly over the last decade. Therefore, LiDAR-based SLAM has been a hot research topic and attracts more and more researchers and entrepreneurs over the past two decades.

LiDAR-based SLAM generally performs a two-stage point cloud registration process, namely, (1) scan-to-scan point cloud registration at a high frequency (for coarse registration) and (2) scan-to-map point cloud registration at a low frequency (for fine registration). Scan-to-scan point cloud registration matches LiDAR-captured point clouds between adjacent frames, which output the robot base's translation and rotation movement between the two frames, while scan-to-map point cloud registration registers the current frame's point clouds to an ever-increasing point cloud map, which, in the meanwhile, get low frequency but high accuracy translation and rotation of the robot base. For the two-stage point cloud registration process (LiDAR odometry and mapping), there are, in general, three categories of methodologies, namely (1) point-based registration, e.g., Iterative Closest Point (ICP) [1]; (2) distribution-based registration, e.g., Normal Distribution Transform (NDT) [2]; and (3) feature-based registration, e.g., LiDAR Odometry And Mapping (LOAM) [3]. A brief literature review of the three categories of point cloud registration methods will be presented in Section II.

Both point-based and distribution-based point cloud registration methods are computationally expensive, and hence are difficult to apply to real-time map construction. While feature-based point cloud registration methods, due to its lightweight feature extraction and matching process, have the potential for real-time map construction. However, when we apply feature-based point cloud registration methods to unstructured environments, sometimes, the extracted features are too sparse, which

Manuscript received 1 May 2021; revised 22 September 2021 and 10 January 2022; accepted 8 February 2022. Date of publication 16 February 2022; date of current version 20 March 2023. *(Corresponding author: Hongliang Guo.)*

Hongliang Guo is with the Institute for Infocomm Research, Agency for Science, Technology and Research, Singapore 138632 (e-mail: guohongliang_uestc@163.com).

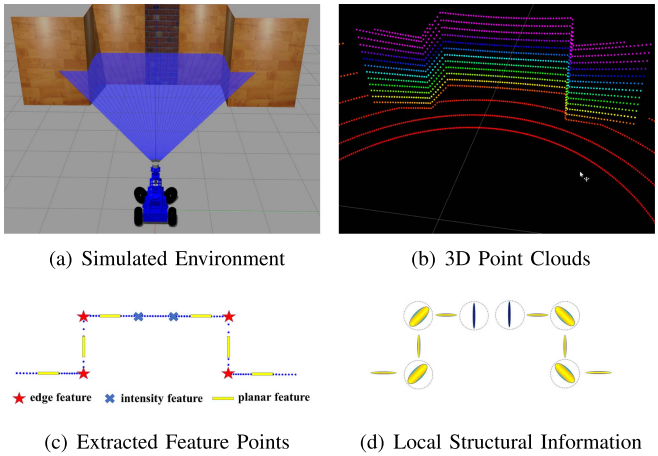
Jiankang Zhu and Yunping Chen are with the School of Automation Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China (e-mail: zhujiankang_chn@foxmail.com; chenyp@uestc.edu.cn).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TIV.2022.3151665.

Digital Object Identifier 10.1109/TIV.2022.3151665

2379-8858 © 2022 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on May 05,2024 at 06:13:56 UTC from IEEE Xplore. Restrictions apply.



(a) Simulated Environment (b) 3D Point Clouds

(c) Extracted Feature Points (d) Local Structural Information

Fig. 1. Example for point cloud's feature space augmentation and local structural information expansion (for illustration purpose only). Fig. 1(a) and Fig. 1(b) show the simulated environment and the corresponding 3D point cloud information, respectively. Fig. 1(c) shows the 2D point cloud and the extracted edge, planar and intensity features, and Fig. 1(d) shows the expanded local structural information approximated by normal distribution.

makes both the scan-to-scan and the scan-to-map point cloud registration process inaccurate, and hence affects the quality of the constructed map.

This paper proposes E-LOAM, which *expands* extracted geometric information with local point clouds around the feature points. We use the normal distribution to represent the local point clouds and thereby capture the local structural information. In this way, we augment the point cloud registration process with additional local structural information around the feature points. With the augmentation, both the scan-to-scan and scan-to-map point cloud registration processes are more accurate, and hence the quality of the constructed map is increased. Additionally, to enrich the sparse features, we make use of the intensity information of the point clouds, and distill the places with high intensity variations as additional feature points. An illustrative example for E-LOAM's feature space augmentation and local structural information expansion is shown in Fig. 1.

The contributions of the paper are summarized as follows: (1) we propose E-LOAM, which augments the point cloud registration process with local structural information around the feature points; (2) we enrich the sparse geometric features in unstructured environments through incorporating the point clouds' intensity information and distilling new features with high intensity variations; and (3) E-LOAM is validated with standard datasets (KITTI odometry benchmark datasets) and also tested on a real robot platform, with satisfying results.

The remainder of the paper is organized as follows: we perform a brief literature review of point cloud registration methods and LiDAR-based SLAM techniques in Section II, followed by a brief introduction of NDT and LOAM in Section III. Then, we present the E-LOAM paradigm in Section IV. Extensive experiments with canonical KITTI odometry benchmark datasets and a real robot platform are presented in Section V, and the paper ends with conclusion and future works in Section VI.

II. Literature Review: LiDAR-Based SLAM

This section presents a brief review of LiDAR-based SLAM methodologies.[1] The essence of LiDAR-based SLAM is to perform a point cloud registration. For point cloud registration, the basis is to establish the correspondences between primitives of the two point clouds whether it is the scan-to-scan stage or the scan-to-map stage. Out of this, an error measurement is derived and minimized. Depending on the types of primitives, we categorize the methodologies for point cloud registration into three groups, namely, (1) point-based registration, (2) distribution-based registration and (3) feature-based registration. Table I provides a bird's-eye view of the mainstream point cloud registration methods.

A. Point-Based Point Cloud Registration

Point-based point cloud registration establishes the correspondence between the target point cloud and the reference point cloud, in the most straightforward way, in which the *points* in the point cloud are the correspondence primitives. The simplest way to find the correspondence points in the reference point cloud is to find the one with the minimal Euclidean distance, i.e., the closest point. Besl and Mckay propose the iterative closest point (ICP) algorithm, which finds the best Euclidean transformation with the minimal least square distances between the point-to-point correspondences [1].

Other researchers make use of the geometric information within the point cloud to restrict the search space, and thereby propose variations of the ICP algorithms [6]–[14]. For example, Behley *et al.* propose SuMa, which constructs the surfel-based map to model the patches of point clouds, and then estimates the changes of robot's pose, i.e., LiDAR odometry, by exploiting the projective data association between the current scan and a rendered model view from the constructed surfel map [11]. The loop closure process can be done in a similar way, which is more robust in terms of overlap detection, than canonical algorithms. In order to improve the point cloud registration accuracy in SuMa, Chen *et al.* propose semantic ICP, which extracts the semantic information from point clouds and removes detected dynamic objects [12]. In general, point-based point cloud registration methods serve as straightforward solutions, and are easy to understand. However, since there are too many points in one frame of the target point cloud, the iterative way of finding all the point-to-point correspondence is time consuming, and thus most ICP algorithms and its variations cannot provide real time scan-to-scan or scan-to-map registration performance.

B. Distribution-Based Point Cloud Registration

Different from point-based registration methods which use points as correspondence primitives, distribution-based registration methods partition the space of point cloud into a set of voxels, and the points within each voxel are approximated with a continuous probability density function (PDF). The correspondence primitives can be deemed as the PDFs within each

[1]Interested readers are referred to [30] for a comprehensive review and [31], [32] for survey of SLAM applications in the autonomous driving domain.

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on May 05,2024 at 06:13:56 UTC from IEEE Xplore. Restrictions apply.

TABLE I
BIRD'S-EYE VIEW OF POINT CLOUD REGISTRATION METHODS

Point-based Registration		Distribution-based Registration		Feature-based Registration	
Canonical	ICP [1]	Canonical	NDT [2]	Canonical	LOAM [3], [4]
Geometric Information Assisted	Point to Line ICP [5] Point to Plane [6], [7] Point to Surface [8], [9] Frame to Mesh ICP [10] Frame to Model ICP [11] Semantic ICP [12] Generalized ICP [13] LiTAMIN [14]	NDT Variations	P2D-NDT [15] D2D-NDT [16], [17]	LOAM Variations	V-LOAM [18] LIO-SAM [19] LeGO-LOAM [20] Semantic LOAM [21], [22]
		Information Assisted NDT	IMU Assisted [23] Emergency Map [24] Occupancy Map [25], [26]	Specific Applications	Small FoV [27] Loop Closure [28] Point to Mesh [29]

voxel, and the objective is to find the Euclidean transformation parameters, which matches the PDFs between the two point clouds the best. Biber proposes the NDT (normal distribution transformation) method for point cloud registration, which uses normal distribution to approximate the points within each voxel, and the best correspondence between two point clouds is defined as the maximizing the probability density score of the reference point cloud given the target point cloud's PDF description [2].

Many variations of NDT are proposed in later threads [15]–[17], [24]. For example, Stoyanov *et al.* propose the distribution-to-distribution NDT method, which describes both the target point cloud and the reference point cloud with discrete sets of PDFs, and the best correspondence is defined as the one with the minimal distance between two distributions, such as the L_2 distance [17]. Other extensions of NDT include fusing IMU sensor for scan-to-scan registration initialization [23], using pre-segmented point clouds to assist the point cloud registration process [33], fusing occupancy map for better scan-to-map registration [25], [26] and fusing the emergency map for better SLAM performance [24]).

When compared with ICP-like methods, NDT and its variations reduce the computation requirements to a certain extent, and capture the structural information through aggregating points within the same voxel together for PDF representation. However, the number of voxels across the whole point cloud space, is still too large, thus, in general, NDT (and its variations) cannot offer the real time LiDAR odometry and mapping functionality when facing large scale applications.

C. Feature-Based Point Cloud Registration

Using raw points (ICP) or voxels (NDT) as correspondence primitives for point cloud registration usually cannot offer real time SLAM functionality, thus, recently, researchers propose to extract representative *feature* points as the correspondence primitives, and then perform scan-to-scan and scan-to-map point cloud registration. Zhang *et al.* propose LOAM (LiDAR odometry and mapping in real time), which extracts the edge and planar features out of the point cloud as the correspondence primitives, and uses the point-to-line and point-to-plane metric as the minimization objective for point cloud registration [3]. LOAM has been considered as the state-of-the-art technique for real time LiDAR-based SLAM in structured environments, where there are many edge and planar features. It is lightweight in terms of computation load, and accurate when there are many salient geometric features, like edges and planes, in the target environment. Other similar features such as curbs [34], lane[35] and facades[36] have also been studied.

Extensions and variations of LOAM have been proposed in the literature [4], [18]–[21], [27], [33]. For example, Shan and Englot propose Lego-LOAM, which uses the ground as the natural information for planar features, and leverages the presence of a ground plane to assist the point-to-plane correspondence process [20]. Lego-LOAM is ground-optimized, and achieves better accuracy with reduced computation expense in most scenarios. Other improvements or variations of LOAM include incorporating vision sensors for additional feature extraction [18], adding loop closure detections for better scan-to-map registration [28], etc.

LOAM and its variants are computationally efficient and thus offer the real time point cloud registration functionality. When applied to structured environments with adequate geometric features, LOAM also performs quite well in terms of map qualities. However, when we are targeting at unstructured environments, where there might not be enough geometric features, such as edges and planes, the accuracy of the self-constructed map decreases greatly. Therefore, we propose E-LOAM, which expands the extracted geometric information by incorporating the local structural information around the extracted feature points. Moreover, E-LOAM enriches the sparse geometric features by including the high intensity variation points as additional features. In this way, E-LOAM augments the feature space for better correspondence in the two-stage point cloud registration process, and in the meanwhile, it is still lightweight, and able to offer the real time map construction functionality.

III. BACKGROUNDS OF LOAM AND NDT

Since the proposed E-LOAM paradigm is based on LOAM [3], and we use normal distribution to approximate the voxels around the pre-extracted feature points, we present LOAM and the normal distribution transform (NDT) methodology in this section as backgrounds for E-LOAM.

A. LiDAR Odometry and Mapping (LOAM)

LOAM is first proposed in [3], which originally targets at the 2D rotational LiDAR scan registration and mapping problem,

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on May 05,2024 at 06:13:56 UTC from IEEE Xplore. Restrictions apply.

but the idea is easily extended to the 3D LiDAR use case. Considering that point-based or distribution-based point cloud registration methods are too time consuming, Zhang and Singh propose to extract representative geometric features out of the raw point clouds as the correspondence primitives [3]. The sharp edges and planar surface patches are the two salient features to be extracted.

Let $\mathcal{P}_k$ be the point cloud perceived at time step k; p_i be a point in $\mathcal{P}_k$, i.e., $p_i \in \mathcal{P}_k$; and S be the set of consecutive points around p_i (which is deemed as p_i's neighbor points within the same laser beam), i.e., $S \subset \mathcal{P}_k$.[2] The following term is defined to gauge the smoothness of the surface around p_i:

$$\Delta c_i = \frac{1}{|S| \, \|p_i\|} \left\| \sum_{p_j \in S, j \neq i} (p_j - p_i) \right\|, \tag{1}$$

where $\|p\|$ refers to the Euclidean norm of a point vector p.

The points in $\mathcal{P}_k$ are sorted according to their Δc_i values, and then feature points are selected with the maximum Δc_i values, namely, edge points, and with the minimum Δc_i values, namely planar points. To evenly distribute the feature points within the point cloud space, $\mathcal{P}_k$ is partitioned into six identical sub-regions. Each sub-region provides at least 2 edge points and 4 planar points. A point p_i is selected as an edge or a planar point only if its Δc_i value is larger or smaller than the respectively predefined threshold.

After the feature points selection, LOAM performs the correspondence process between two adjacent LiDAR point clouds, $\mathcal{P}_{k-1}$ and $\mathcal{P}_k$. The key idea is to find the Euclidean transform (rotation and translation) which minimizes the point-to-line distance for edge points, and minimizes the point-to-plane distance for planar points. Interested audiences are referred to [3] for details.

B. Normal Distributions Transform (NDT)

NDT is a distribution-based point cloud registration method, originally proposed in [2]. NDT partitions the point cloud into a discrete set of voxels, and the points within each voxel are approximated by a multivariate normal distribution. The size of the voxels is normally set to be small for indoor environments (e.g., $0 \sim 2$m per edge) and large for outdoor environments(e.g., $2 \sim 10$m per edge). Let $V_i \subset \mathcal{P}_k$ denote a voxel of the point cloud,[3] and the parameters of the approximated multivariate normal distribution are estimated as follows:

$$\boldsymbol{\mu}_i = \frac{1}{|V_i|} \sum_{p \in V_i} p, \tag{2}$$

$$\boldsymbol{\Sigma}_i = \frac{1}{|V_i|} \sum_{p \in V_i} (p - \boldsymbol{\mu}_i)(p - \boldsymbol{\mu}_i)^\top, \tag{3}$$

where $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$ are the mean and covariance of the approximated multivariate normal distribution for V_i.

[2]Here, we wish to note that the LiDAR sensor used in this paper is a mechanical rotating LiDAR with different laser beams. Therefore, S represents continuous points within each laser beam, which can be regarded as an ordered set of points.

[3]Note that V_i represents the i^{th} voxel in the three-dimensional space, ignoring the influence of the LiDAR working mechanism, and thus it can be considered as a disordered point set

In the following, we describe the correspondence process of a specific NDT method, namely, distribution-to-distribution NDT (D2D-NDT) [17], as E-LOAM uses the similar correspondence process to match the local structural information between adjacent point cloud frames. Defining the rotation matrix as $\boldsymbol{R}$ and translation vector as $\boldsymbol{t}$, the correspondence process of D2D-NDT is to find the 'best' $\boldsymbol{R}$ and $\boldsymbol{t}$ values, which minimize the distribution distance (e.g., the L_2 distance) between the transformed normal distribution for $\mathcal{P}_{k-1}$ and the approximated normal distribution for $\mathcal{P}_k$. Detailed problem formulation and optimization process can be referred in [17].

IV. E-LOAM: LiDAR Odometry and Mapping With Expanded Local Structural Information

This section presents the complete E-LOAM paradigm, including (1) geometric and intensity feature extraction and matching; (2) local structural information representation with normal distribution transform; (3) map construction and pose optimization; and (4) loop detection and closure.[4] The pipeline of E-LOAM is visualized in Fig. 2.

A. Geometric and Intensity Feature Extraction and Matching

Given a 3D point cloud $\mathcal{P}_k$, E-LOAM employs the same method as in [3] to extract the *edge points* (points with large Δc_i values in (1)) and *planar points* (points with small Δc_i values in Eq. (1)) as geometric features. However, in some unstructured environments, the salient geometric features, i.e., edge points and planar points, might be sparse. Therefore, in E-LOAM, we augment the feature points by considering the intensity information of the point cloud. Before presenting the intensity feature extraction process, we display Fig. 3 as an illustrative example for point cloud's extracted features, i.e., planar feature, edge feature, intensity feature, in E-LOAM.

For each point $p_i \in \mathcal{P}_k$, besides the 3D coordinate information (which is naturally encoded in p_i), LiDAR also returns the intensity information of p_i, denoted as I_i. Small I_i means that p_i is either far from the LiDAR sensor or the object's reflectance is low. We calculate the intensity 'variance' (ΔI_i) of p_i as follows:

$$\Delta I_i = \frac{1}{|S|} \sum_{p_j \in S, j \neq i} (I_j - I_i)^2, \tag{4}$$

If ΔI_i is larger than a predefined threshold value, then p_i is deemed as an additional feature point with high intensity variation. To evenly distribute the intensity feature points within the point cloud space, $\mathcal{P}_k$ is partitioned into several identical sub-regions, with each sub-region providing a fixed number of intensity feature points. Here, we wish to note that the intensity value measurement always contains noise. If Eq. (4) is applied directly to the raw intensity inputs for intensity feature extraction, there will be a lot of noise points extracted as features. In practice, we use the mean filter to pre-process the raw LiDAR intensity data. In this way, most of the noise will be filtered out. An illustrative example is presented in Fig. 4, which shows how

[4]Loop detection and closure is an add-on module, without which, E-LOAM and other LiDAR-based SLAM methods are still functioning. The benefits of loop detection and closure is to make the self-constructed map more accurate.

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on May 05,2024 at 06:13:56 UTC from IEEE Xplore. Restrictions apply.

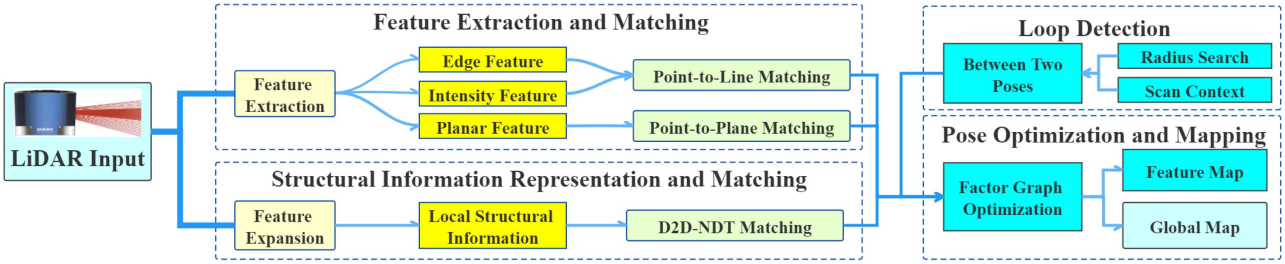
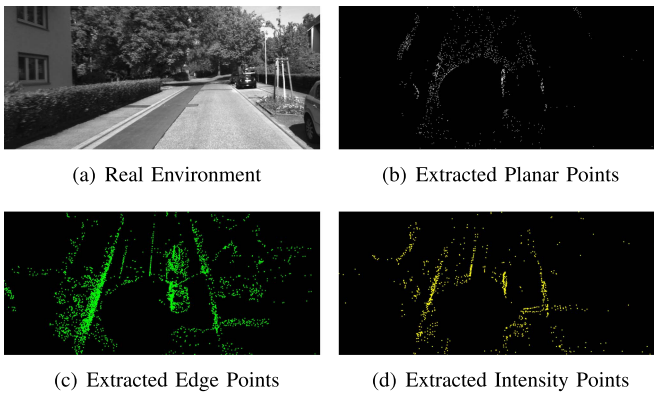


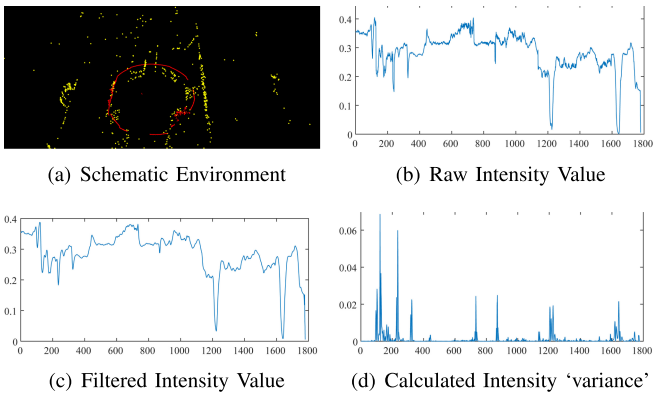
Fig. 2. E-LOAM Pipeline: E-LOAM takes the 3D point cloud as input; extracts both geometric features and intensity-related features; expands the features with local structural information; and performs LiDAR odometry and mapping with the help of loop detection and closure.



(a) Real Environment (b) Extracted Planar Points

(c) Extracted Edge Points (d) Extracted Intensity Points

Fig. 3. Example for point cloud's feature extraction. Fig. 3(a) shows the real environment. Fig. 3(b), (c), and (d) show the extracted planar points, edge points and intensity points, respectively.



(a) Schematic Environment (b) Raw Intensity Value

(c) Filtered Intensity Value (d) Calculated Intensity 'variance'

Fig. 4. Example of the intensity feature extraction process. Fig. 4(a) shows the schematic environment, the points in green represent extracted intensity features and the points in red represent one beam of LiDAR raw points. Fig. 4(b) and Fig. 4(c) show the raw intensity value and mean filtered intensity value, respectively. Fig. 4(d) shows the intensity 'variance' calculated by Eq. (4).

the mean filter works to 'smooth' the intensity data, and then the intensity features are extracted on top of the smoothed data.

We argue that the stated way of intensity-related feature extraction is reasonable, in that p_i with a high intensity variation, means that there are, most likely, material changes near the point. For example, the material around p_i changes from wood to metal, which has a different reflectance ability, and in this case, p_i will have a high intensity variation value. With previously extracted geometric features, the augmented feature space constitutes three types of feature points, namely, edge points, planar points, and intensity points.

Next, we will perform the matching process of pre-extracted feature points between two point clouds. Let $\mathscr{P}$ denote the set of extracted feature points (edge points, planar points, and intensity points). For point cloud $\mathcal{P}_{k-1}$, each feature point $p_i \in \mathscr{P}$, we perform Euclidean transformation based on $\boldsymbol{R}$ and $\boldsymbol{t}$, i.e., $p_i' = \boldsymbol{R}p_i + \boldsymbol{t}$. If p_i is an edge point or an intensity point,[5] we calculate the point-to-line distance to the correspondence in $\mathcal{P}_k$ as the objective:

$$d_{\mathcal{L}}^{(i)} = \frac{\|(p_i' - p_{j1}) \times (p_i' - p_{j2})\|}{\|p_{j1} - p_{j2}\|}, \tag{5}$$

where $p_{j1}, p_{j2} \in \mathcal{P}_k$ are the correspondence primitives for $p_i \in \mathcal{P}_{k-1}$, and $p \times q$ refers to the cross product of the two vectors p and q. On the other hand, if p_i is a planar point, we calculate the point-to-plane distance as the objective:

$$d_{\mathcal{P}}^{(i)} = \frac{\|(p_i' - p_{j1})(p_{j1} - p_{j2}) \times (p_{j1} - p_{j3})\|}{\|(p_{j1} - p_{j3}) \times (p_{j2} - p_{j3})\|}, \tag{6}$$

where $p_{j1}, p_{j2}, p_{j3} \in \mathcal{P}_k$ are the correspondence primitives for $p_i \in \mathcal{P}_{k-1}$.

B. Local Structural Information Representation With Normal Distribution Transform

In the previous subsection, we augment the feature space with intensity-related feature points. However, both intensity and geometric features are *pointed* features, which represent the information of a certain point. In most use cases, the local structural information is also important. This subsection introduces a feature space expansion method which represents the local structural information around the pre-extracted feature points.

Given a feature point $p_i \in \mathcal{P}_k$, we construct a voxel $V_i \subset \mathcal{P}_k$. V_i is a subset of points, which fall inside a sphere centered at p_i, with radius r (e.g., $2m \leq r \leq 10m$). Specifically, we have:

$$V_i = \{p\,;\, p \in \mathcal{P}_k, \|p - p_i\| \leq r\}. \tag{7}$$

In E-LOAM, the probability density of V_i is approximated by a multi-variate normal distribution, i.e., $\mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$. The related parameters are calculated as:

$$\boldsymbol{\mu}_i = \frac{1}{|V_i|} \sum_{p \in V_i} p, \tag{8}$$

[5]Note that, in essence, the intensity point with high intensity variation can be deemed as an 'edge' point in the intensity domain.

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on May 05,2024 at 06:13:56 UTC from IEEE Xplore. Restrictions apply.

$$\boldsymbol{\Sigma}_i = \frac{1}{|V_i|} \sum_{p \in V_i} (p - \boldsymbol{\mu}_i)(p - \boldsymbol{\mu}_i)^\top. \tag{9}$$

For the structural correspondence process, the objective is to minimize the L_2 distance between the two normal distribution approximations of $\mathcal{P}_{k-1}$ and $\mathcal{P}_k$. The structural correspondence objective is expressed as:

$$f_{\text{d2d}} = \sum_{V_i \in \mathcal{P}_{k-1}} \sum_{V_j \in \mathcal{P}_k} -d_1 \exp\left(-\frac{d_2}{2} \boldsymbol{\mu}_{ij}^\top (\boldsymbol{R}^\top \boldsymbol{\Sigma}_i \boldsymbol{R} + \boldsymbol{\Sigma}_j)^{-1} \boldsymbol{\mu}_{ij}\right), \tag{10}$$

where $\boldsymbol{\mu}_{ij} = \boldsymbol{R}\boldsymbol{\mu}_i + \boldsymbol{t} - \boldsymbol{\mu}_j$ and d_1, d_2 are regularization factors.

Denote $0 \leq \theta \leq 1$ as a weight parameter balancing the relative importance between point correspondence and structural correspondence, we have the LiDAR odometry objective as:

$$\text{obj}(\boldsymbol{R}, \boldsymbol{t}) = \theta \sum_{p_i \in \mathscr{P}} \left(d_{\mathcal{L}}^{(i)} + d_{\mathcal{P}}^{(i)}\right) + (1 - \theta) f_{\text{d2d}}. \tag{11}$$

With any designated θ, and the standard optimization solver, we can find the optimal $\boldsymbol{R}$ and $\boldsymbol{t}$ as the motion estimation of the robot base.

C. Pose Optimization and Map Construction

So far, we have extracted the geometric and intensity features out of point clouds, and augmented the features with local structural information (approximated by NDT). In this subsection, we perform robot's pose sequence optimization and map construction.

During the point cloud registration process, one can get the robot base's pose for each frame. However, as the process continues, the sequence of robot poses inevitably drifts with accumulated errors. Therefore, we need to optimize the robot pose sequence after a certain period. In E-LOAM, we use the factor graph to correlate each frame and build Bayesian networks for joint representation. By maximizing the posterior distribution of robot poses, we achieve more accurate estimations. In practice, we use the GTSAM open source library [37] to optimize the robot poses.

The map construction process focuses on matching the point cloud $\mathcal{P}_k$ with a map database $\mathcal{M}_{k-1}$, which is a history collection of (transformed) point clouds up to time step $k-1$. Since the map $\mathcal{M}_{k-1}$ contains almost the whole point cloud history information, it is much denser in terms of point cloud densities than a single point cloud scan. The goal of LiDAR Mapping is to find the best Euclidean transformation parameters which best map $\mathcal{P}_k$ to $\mathcal{M}_{k-1}$, and hence produce $\mathcal{M}_k$ by incorporating $\mathcal{P}_k$.

D. Loop Closure Detection and Optimization

The LiDAR odometry error is prone to accumulate and result in the map ($\mathcal{M}_k$)'s drift and structural error after the robot base's long-distance travel [28]. In order to avoid such problems, it is necessary to associate closed-loop data and correct the corresponding LiDAR odometry sequences. Recently, there is a growing trend of using deep neural networks (DNNs) to recognize the similarity of two frames of point clouds [38], and such methods achieve high accuracy results in loop detection. However, DNN-based methods require a large amount of calculation and GPU acceleration. Since our goal is to achieve a lightweight SLAM system, in E-LOAM, we simply calculate the similarity score between two frames based on the Euclidean distance and scan context [39]. When the similarity scores of both the Euclidean distance and scan context are less than the pre-defined threshold, the relative pose will be calculated by ICP registration and added to the factor graph optimization process. We follow the standard procedure, e.g., the GTSAM library, to optimize the related parameters and hence reduce the map's drift and structural error. Thanks to the robust kernel in GTSAM, the system will not fail even if there is a false positive cycle factor in the pose graph.

V. Experimental Results and Analysis

In this section, we evaluate and compare the performance of E-LOAM with five state-of-the-art algorithms, namely LOAM [3], LeGO-LOAM [20], HDL-NDT(HDL-Graph-SLAM) [40], SuMa [11] and SuMa++ [12]. LOAM is a feature-based LiDAR odometry and mapping system which operates in real time. LeGO-LOAM is a lightweight and ground-optimized algorithm of LOAM [3] with the ground-plane optimization, and HDL-NDT (High Definition LiDAR Normal Distributions Transform) realizes the real-time operation ability of the canonical NDT method [2] with multi-thread operation (The canonical NDT algorithm cannot offer real-time LiDAR-based SLAM service). SuMa uses Surfel maps to implement front-end odometry and loop closure, and SuMa++ uses semantic segmentation to remove dynamic obstacles and associate semantic data on this basis. Note that this paper uses LiDAR as the only input sensor. Therefore, for multi-sensor input algorithms, e.g., LOAM, LeGO-LOAM and HDL-NDT, the experimental results of this paper are worse than those reported in original paper.

All the algorithms are evaluated on a PC with 2.5GHz quad cores, 1050Ti GPU, 16GB RAM and robot operating system (ROS) [41] running in Ubuntu 18.04. The software, coded in C++, for E-LOAM, and detailed configuration parameters (e.g., θ), are publicly available.[6] In the following three subsections, we first quantitatively compare E-LOAM with the aforementioned state-of-the-art algorithms for the KITTI odometry benchmark datasets [42] and then analyze E-LOAM's performance qualitatively on a real robotic platform. In the end, we evaluate and compare E-LOAM with state of the arts in some extreme use cases, i.e., feature degenerative scenarios and dynamic environments. Feature degenerative scenarios refers to the environments with too few features to be extracted, and dynamic scenarios refer to the scenario with a lot of moving objects.

A. Quantitative Comparison With KITTI Odometry Datasets

This subsection evaluates the performance of E-LOAM, LOAM, LeGO-LOAM, HDL-NDT, SuMa and SuMa++ with the KITTI odometry benchmark datasets, which have been carefully registered with sensors mounted on top of a passenger vehicle driving on structured roads. The vehicle is equipped with a 360°

[6] https://github.com/zhujiankang/E-LOAM

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on May 05,2024 at 06:13:56 UTC from IEEE Xplore. Restrictions apply.

TABLE II
QUANTITATIVE RESULTS FOR KITTI ODOMETRY SEQUENCES 00-10

Seq. num.	Environment	E-LOAM	LOAM	LeGO-LOAM	HDL-NDT	SuMa	SuMa++
00	urban	0.0047/1.1722	0.0059/1.3373	0.0086/1.7650	0.0083/1.2885	0.0050/1.0909	**0.0031/0.7470**
01	highway	0.0062/2.9205	0.0067/3.0459	0.0093/14.3887	0.0090/58.3958	0.0106/14.7107	**0.0052/2.1157**
02	urban	0.0082/2.3621	0.0157/5.0031	0.0111/3.0724	0.0178/8.0862	0.0130/4.4661	**0.0041/1.3821**
03	country	**0.0068**/1.1558	0.0072/1.5546	0.0099/1.5995	0.0078/1.5401	0.0077/**1.0663**	0.0086/1.7593
04	highway	0.0050/1.3885	0.0047/1.3917	0.0084/1.8219	0.0054/1.8347	0.0057/1.4449	**0.0031/0.4649**
05	urban	0.0036/0.8198	0.0042/0.8800	0.0070/1.3804	0.0087/1.4410	0.0042/0.8650	**0.0024/0.5066**
06	urban	0.0057/1.3350	0.0056/1.2931	0.0060/1.3851	**0.0036/0.5421**	0.0036/0.6770	0.0038/1.2274
07	urban	0.0066/1.2051	0.0041/0.6476	0.0081/1.3880	0.0148/1.8503	0.0076/1.5037	**0.0030/0.3589**
08	country	0.0063/1.6271	0.0052/1.2729	0.0080/1.7849	0.0101/2.0502	0.0054/1.2546	**0.0037/1.0602**
09	country	0.0057/1.3587	0.0069/1.7571	0.0085/1.9340	0.0091/1.9958	0.0062/1.3897	**0.0047/1.1192**
10	country	0.0063/1.8404	0.0065/1.9745	0.0128/2.7367	0.0127/3.0633	0.0077/2.1250	**0.0060/1.7624**
Average Performance		0.0059/1.5623	0.0066/1.7533	0.0089/3.0233	0.0098/7.4625	0.0070/2.7804	**0.0043/1.1367**

Note that the performance is represented in 'A/B' format, where 'A' indicates the relative rotational error in degrees per meter, and 'B' indicates the relative translation error in %. Bold numbers means top performance out of the six evaluated algorithms.

Velodyne LiDAR, and a high accuracy GPS/INS for ground truth purposes. The datasets mainly cover three types of environments: (1) 'urban' with buildings around; (2) 'country' on small roads with vegetations in the scene; and (3) 'highway' where roads are wide and the surrounding environment is relatively clean. The overall driving distance included in the datasets is 39.2 km.

Table II shows the comparative performance of E-LOAM, LOAM, LeGO-LOAM, HDL-NDT, SuMa and SuMa++. In E-LOAM, LOAM and LeGo-LOAM, we employ both scan-to-scan and scan-to-map registration methods. The operating frequency of the scan-to-scan method is 10 Hz (real time), and the operating frequency of the scan-to-map method including the robot pose optimization is 5 Hz. The performance is evaluated with the finally optimized robot pose sequences.

In the table, we can see that both E-LOAM consistently outperforms LOAM and LeGO-LOAM in terms of the average translation error and average rotation error for most sequences. The observation is reasonable, in that E-LOAM augments the original feature space of LOAM with expanded local structural information, moreover, E-LOAM further takes the intensity-related information into consideration. It should be noted that in Table II, the accuracy of LOAM is worse than that reported on KITTI odometry leaderboard. There are two reasons for it: (1) since we are targeting a real-time SLAM service, we select the canonical LOAM [3] as the baseline, while the LOAM results reported on KITTI odometry leaderboard is not a real-time operation, instead, it is processed at 10% of the real-time speed, taking one second to process a scan, see Section 7.4 of [4] for details; (2) the LOAM results reported on KITTI odometry leaderboard are obtained with an additional sensor input (IMU) for better accuracy, since we are targeting the LiDAR-only SLAM, we did not use IMU in our LOAM version. For LeGO-LOAM, although it is deemed as an improved version of LOAM, its main focus is for the lightweight and real-time LiDAR odometry. The main reason that our reported error of LeGO-LOAM is larger than the results obtained in the original paper [20] is that in our LeGO-LOAM implementation, we use only one sensor, i.e., LiDAR, for comparison fairness. While the reported results in [20] also use IMU as the additional sensor. For HDL-NDT, since essentially it performs normal distribution approximation for all the voxels, its accuracy should be higher than all the LOAM-related algorithms, which only consider the feature points. However, in cases where there are enough feature points (such as Sequence 00, 01, 02 in the table), E-LOAM outperforms HDL-NDT in terms of both translation error and rotation error. An additional advantage of E-LOAM when compared with HDL-NDT is its lower computational cost, which results in less needed computation time for both scan-to-scan registration and scan-to-map registration. The reason is that E-LOAM only employs normal distribution approximation around the feature points, while HDL-NDT performs normal distribution approximation all over the space.

When comparing with SuMa and SuMa++, we find that E-LOAM performs better than SuMa in almost all evaluated KITTI data sequences. However, its accuracy is, in most cases, lower than SuMa++. The reason is that SuMa++ further uses deep learning to segment the semantic point cloud on the basis of SuMa and then uses the semantic point cloud for registration (in the process, the semantic point cloud is further filtered to remove dynamic objects), which achieves higher accuracy. Here, we wish to note that SuMa++ incurs the deep learning procedure, which needs the tedious pre-training process and requires GPU for computation acceleration. Therefore, it is quite difficult for SuMa++ to operate in real time, i.e., operate in mili-seconds.

E-LOAM proposes point cloud registration process with both the local structural information around the feature points and intensity feature points to enrich the sparse geometric features. In order to clarify the contribution of the two parts to the algorithm, we use the same KITTI dataset to conduct the ablation study and show the comparative results in Table III. As it can be seen from the table, with the two co-dependent functional modules, the E-LOAM algorithm has best performance.

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on May 05,2024 at 06:13:56 UTC from IEEE Xplore. Restrictions apply.

TABLE III
ABLATION STUDY OF E-LOAM

Seq. num.	E-LOAM(a)	E-LOAM(b)	E-LOAM(a+b)
00	0.0072/1.6517	0.0099/1.4758	0.0047/1.1722
01	0.0061/2.8534	0.0060/2.8076	0.0062/2.9205
02	0.0092/2.8177	0.0140/3.1112	0.0082/2.3621
03	0.0078/1.5283	0.0104/1.4437	0.0068/1.1558
04	0.0060/1.6169	0.0097/1.1967	0.0050/1.3885
05	0.0035/0.9172	0.0051/0.7319	0.0036/0.8198
06	0.0040/0.7682	0.0083/1.2799	0.0057/1.3350
07	0.0043/0.9134	0.0106/1.2729	0.0066/1.2051
08	0.0051/1.3195	0.0099/2.1616	0.0063/1.6271
09	0.0065/1.5646	0.0064/1.0943	0.0057/1.3587
10	0.0075/1.9398	0.0084/1.5986	0.0063/1.8404
Avg.	0.0061/1.6264	0.0090/1.6522	0.0059/1.5623

Note that E-LOAM(a) refers to the E-LOAM version with only the added intensity feature, E-LOAM(b) refers to the E-LOAM version with only the expanded structural information and E-LOAM(a+b) refers to the E-LOAM version with both information.

TABLE IV
RUN TIME AND CPU OCCUPANCY COMPARISON

Methods	Scan Matching(ms)	Mapping(ms)	CPU(%)
LOAM	48.0000	201.0000	20.00
LeGO-LOAM	**7.9442**	**65.7548**	**9.05**
E-LOAM	45.2527	77.3757	24.17
HDL-NDT	148.1187	272.0785	37.50

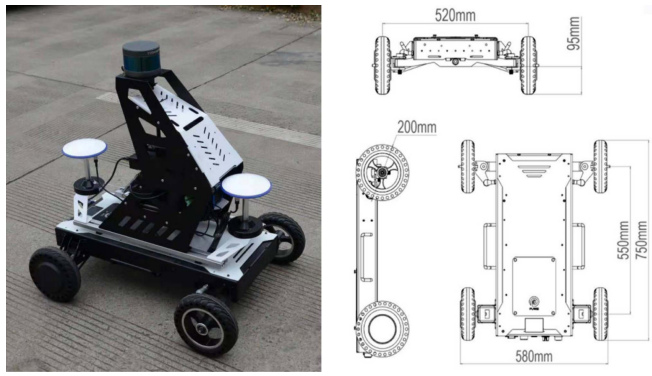
Note that both SuMa and SuMa++ require GPU to accelerate the computation process and SuMa++ has huge overhead computation load for the semantic feature extraction, and thus the related run time statistics and CPU occupancy percentage cannot be measured.

Table IV shows the comparative results of E-LOAM and three benchmark algorithms' average computation time for scan matching, mapping as well as CPU occupancy percentage. Since the all the algorithms are based on ROS and have several ROS nodes, when calculating scan-matching and mapping time, only the related ROS nodes are calculated. In the table, we can see that LOAM, LeGO-LOAM and E-LOAM consume less time than that of HDL-NDT.[7] On the other hand, when we compare the computation time of E-LOAM with that of LeGO-LOAM, we can see that they are more or less at the same scale (less than 0.1 seconds for both scan-to-scan matching and scan-to-map registration for all the three algorithms.). In summary, with the previously specified hardware configuration, E-LOAM ranks the second (behind SuMA++) in terms of the average accuracy of robot's pose estimation, out of the evaluated benchmark algorithms, but it offers real time SLAM performance, i.e., reacts in mili-seconds, which SuMa++ cannot provide.

B. Qualitative Evaluation With Real Robot Platform

In this subsection, we evaluate and compare the performance of E-LOAM with LOAM, LeGO-LOAM and HDL-NDT on a real robot platform, with a RoboSense RS-LiDAR-16, mounted on a wheeled robot base, as shown in Fig. 5. Here, we wish to point out that, for the real world experiment, we do not compare E-LOAM with SuMa or SuMa++, in that currently SuMa and SuMa++ only support the KITTI data format; moreover, SuMa++ needs to re-train the deep segmentation network for any other scenario beyond KITTI dataset. We collect three different point cloud datasets and name them as *Campus*, *Forest*, *Buildings*, which, according to their names, are collected in different types of environments. The details of the datasets are shown in Table V and the datasets collection environments are visualized in Fig. 6. Note that when collecting the data, the starting point and ending point of the robot are the same, which means that the robot's trajectory is actually a loop.

[7]Note that in general, real-time localization applications (especially the autonomous driving related application), require that the scan-to-scan matching time is less than 0.1 seconds, while the scan-to-map registration process can be tolerated to some extent.



(a) Robot Testbed (b) Size Information

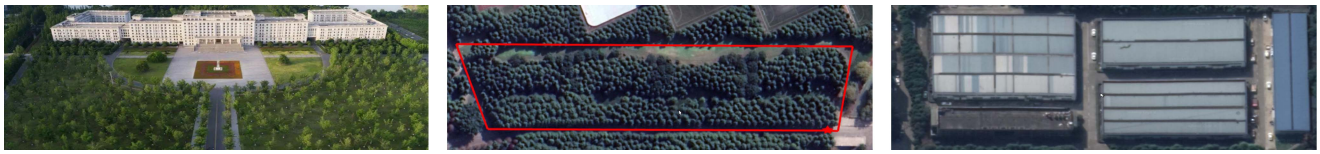
Fig. 5. Real robot testbed and its size information.

TABLE V
DATASET DETAILS

Dataset	# of Scans	Trajectory length(m)	Max Speed(m/min)
Campus	6829	1247	57.6
Forest	4666	702	18
Buildings	6374	427	10.8

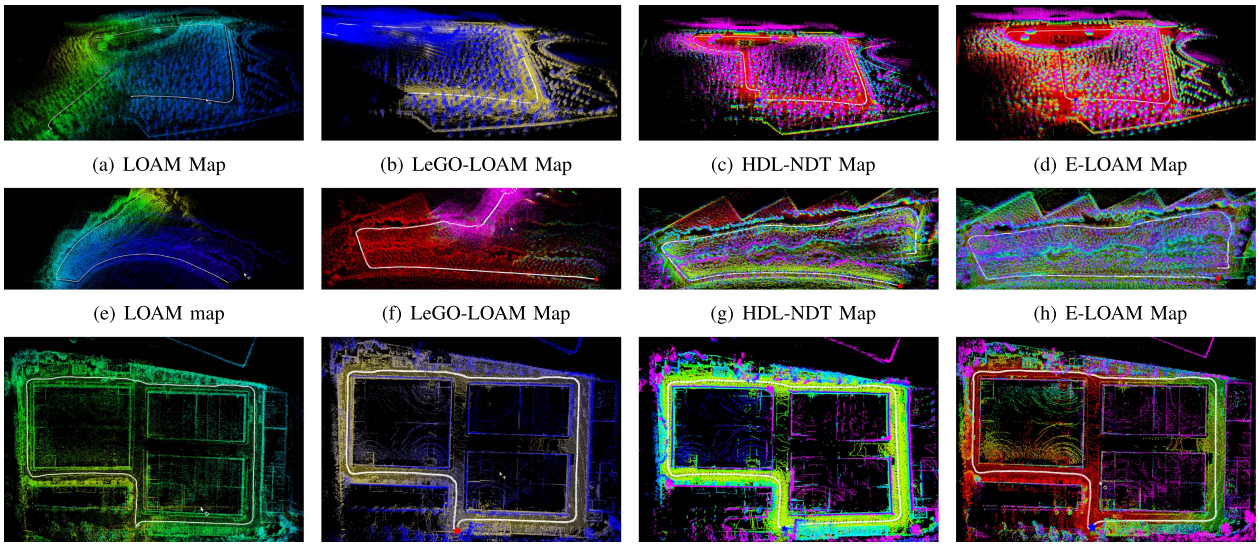
Fig. 7 shows the mapping results of LOAM, LeGO-LOAM, HDL-NDT and E-LOAM for the three datasets. From the figure, the following observations and explanations are drawn. (1) In the *Campus* and *Forest* environments, LOAM and LeGO-LOAM fail to construct a proper map, the LiDAR odometry for LOAM and LeGO-LOAM drifts badly during the map construction process, and the robot is completely lost, half way during navigation. We believe that the underlying reason is that there are too few salient geometric features for LOAM and LeGO-LOAM to extract and correspond. (2) HDL-NDT and E-LOAM perform quite well for all the three environments, but we can still visibly observe from the bottom right corner of Fig. 7(g) and Fig. 7(h) that for the *Forest* environment, E-LOAM has a much smaller odometry drift than that of HDL-NDT (similar phenomenon can be observed in the *Campus* environment as well). This phenomena indicates that it might not be necessary to construct the normal distribution approximation for all the LiDAR scan points, instead, focusing on the distribution approximation around salient feature points yields even better mapping performance, i.e., with lower odometry drifts. (3) All the four algorithms, especially LeGO-LOAM, perform

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on May 05,2024 at 06:13:56 UTC from IEEE Xplore. Restrictions apply.



(a) Campus (b) Forest (c) Building

Fig. 6. Real Environment image of Campus, Forest and Buildings.

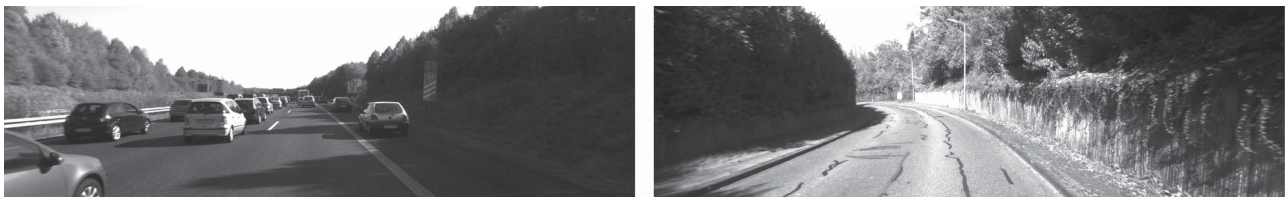


(a) LOAM Map (b) LeGO-LOAM Map (c) HDL-NDT Map (d) E-LOAM Map

(e) LOAM map (f) LeGO-LOAM Map (g) HDL-NDT Map (h) E-LOAM Map

(i) LOAM Map (j) LeGO-LOAM Map (k) HDL-NDT Map (l) E-LOAM Map

Fig. 7. Visual comparison of constructed maps from different algorithms: LOAM, LeGO-LOAM, HDL-NDT, E-LOAM+, SuMa and SuMa++.



(a) Dynamic Environment (b) Degenerative Environment

Fig. 8. Two representative special scenarios.

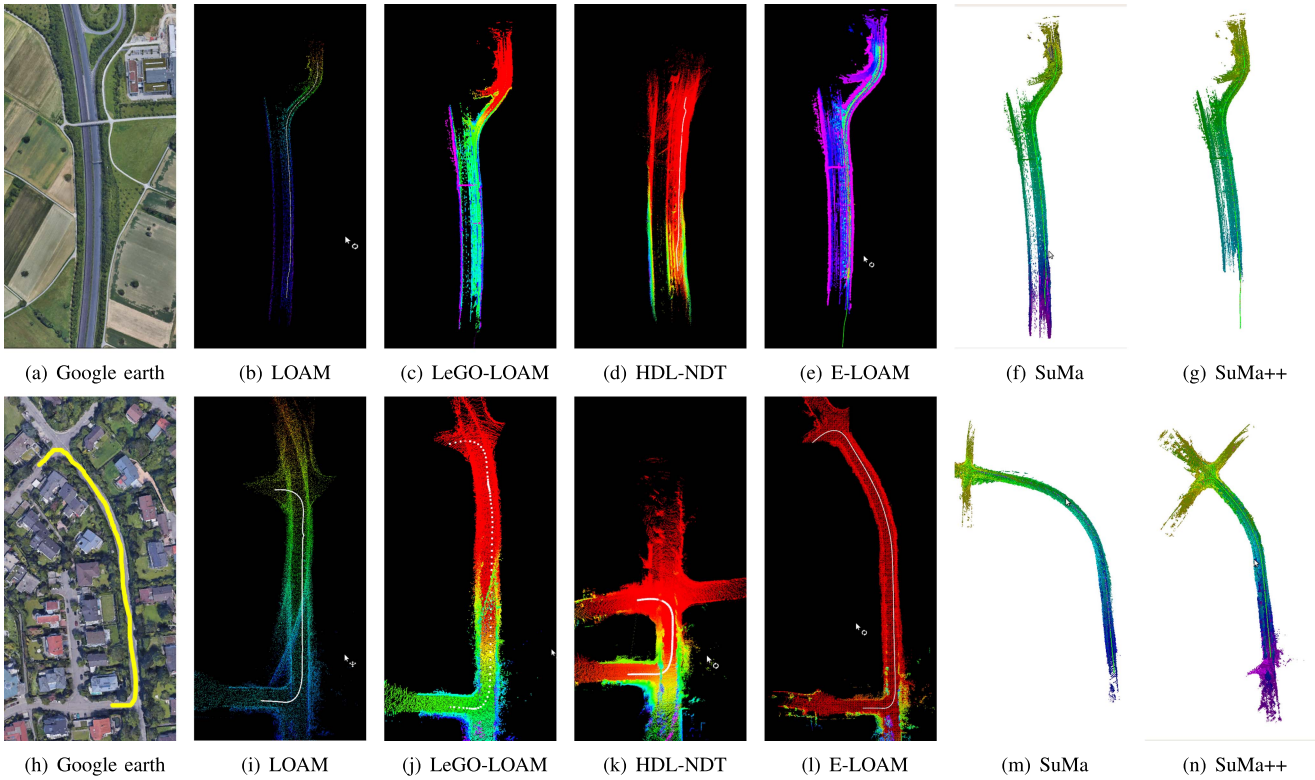
quite well in the *Buildings* environment, in that the *Buildings* environment offers enough geometric features (both edge and planar features) to assist the map construction process in LOAM, LeGO-LOAM and E-LOAM. The scan-to-scan point cloud registration time is, on average, 0.05 seconds for LOAM, E-LOAM and LeGO-LOAM, which can be deemed as real-time LiDAR odometry. For HDL-NDT, the scan-to-scan point cloud registration time is around 0.3 seconds, which is not enough for high speed SLAM scenario requirements, e.g., the *Campus* environment, whose data collection speed reaches about 1 m/s at the maximum.

C. Qualitative Evaluation for Special Scenarios

So far, we have evaluated and compared E-LOAM's performance in the canonical KITTI odometry dataset and three real world scenarios. In this subsection, we are interested in evaluating E-LOAM's performance in special scenarios, e.g., (1) dynamic environment: scenario with a lot of dynamic objects, (2) degenerative environment: scenario with many self-similar structures. We select two representative scenarios (shown in Fig. 8) out of the KITTI odometry dataset, and evaluate the mapping results of E-LOAM and other state of the arts.

Fig. 9(b) to Fig. 9(g) show the comparative results of constructed maps for the dynamic environment from different SLAM algorithms, namely, E-LOAM, LOAM, LeGO-LOAM, HDL-NDT, SuMa and SuMa++. In the figure, we can see that all the algorithms except for SuMa++, fail to construct a high-precision map. The underlying reason is that there are many moving objects in the environment, which makes both scan-to-scan and scan-to-map point cloud registration process error prone. While SuMa++ has an extra functional module, which removes the moving objects in the dynamic environments

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on May 05,2024 at 06:13:56 UTC from IEEE Xplore. Restrictions apply.



(a) Google earth (b) LOAM (c) LeGO-LOAM (d) HDL-NDT (e) E-LOAM (f) SuMa (g) SuMa++

(h) Google earth (i) LOAM (j) LeGO-LOAM (k) HDL-NDT (l) E-LOAM (m) SuMa (n) SuMa++

Fig. 9. Visual comparison of constructed maps from different algorithms: LOAM, LeGO-LOAM, HDL-NDT, E-LOAM, SuMa and SuMa++ (Fig. 9(a) and Fig. 9(h) show the corresponding Google earth maps).

through semantic segmentation. In this way, SuMa++ functions well in the dynamic scene environments.

For the degenerative environment, we show the comparative results from Fig. 9(i) to Fig. 9(n). From Fig. 9(i) tp Fig. 9(k), we can see that LOAM, LeGO-LOAM and HDL-NDT all have severe curvature drifts of the final constructed map, due to too many self-similar structures. LOAM, LeGO-LOAM and HDL-NDT all fail to capture the actual curvature information in the environment. On the other hand, E-LOAM, SuMa and SuMa++ are able to capture the curvature information, but SuMa has larger registration error than the other two algorithms. We believe that the underlying reason of E-LOAM's better performance in degenerative environment is that it incorporates the intensity feature and local structural information for better correspondence.

VI. Conclusion and Future Works

This paper presents E-LOAM, which expands the feature points with local structural information and augments the feature space with intensity-related features. We argue that, in outdoor unstructured environments, where there are few salient geometric features, E-LOAM is able to capture more information for LiDAR odometry and mapping. Experimental results with both KITTI odometry datasets and self-collected real-world datasets, demonstrate the efficacy and efficiency of E-LOAM. The performance of E-LOAM in dynamic environments and degenerative environments are also tested and compared with state of the arts.

In the future, we would like to extend E-LOAM to include vision-based features as well as the expanded local textural information to enhance its mapping functionality. We are also keen on increasing E-LOAM's robustness in dynamic environments with many moving objects. In the meanwhile, we plan to implement E-LOAM with on-board processors, so that E-LOAM can be embedded into real robotic platforms for real-time LiDAR odometry and mapping services. Another interesting future direction is to propose an automatic feature reduction scheme to reduce the overlap of extracted features of E-LOAM, as the intensity feature might overlap with the edge feature, in many real world use cases.

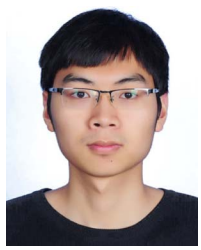
References

[1] P. Besl and N. D. McKay, "A method for registration of 3-D shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 239–256, Feb. 1992.
[2] P. Biber and W. Straßer, "The normal distributions transform: A new approach to laser scan matching," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2003, pp. 2743–2748.
[3] J. Zhang and S. Singh, "LOAM: LiDAR odometry and mapping in real-time," *Proc. Robot., Sci. Syst.*, vol. 2, no. 9, pp. 1–9, Jul. 2014.
[4] J. Zhang and S. Singh, "Low-drift and real-time LiDAR odometry and mapping," *Auton. Robots*, vol. 41, no. 2, pp. 401–416, 2017.
[5] A. Censi, "An ICP variant using a point-to-line metric," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2008, pp. 19–25.
[6] S.-Y. Park and M. Subbarao, "An accurate and fast point-to-plane registration technique," *Pattern Recognit. Lett.*, vol. 24, no. 16, pp. 2967–2976, 2003.
[7] N. Rufus, U. K. R. Nair, A. S. B. Kumar, V. Madiraju, and K. M. Krishna, "SROM: Simple real-time odometry and mapping using LiDAR data for autonomous vehicles," in *Proc. IEEE Intell. Veh. Symp.*, 2020, pp. 1867–1872.

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on May 05,2024 at 06:13:56 UTC from IEEE Xplore. Restrictions apply.

[8] J. Serafin and G. Grisetti, "NICP: Dense normal based point cloud registration," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 742–749.
[9] J.-E. Deschaud, "IMLS-SLAM: Scan-to-model matching based on 3D data," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 2480–2485.
[10] I. Vizzo, X. Chen, N. Chebrolu, J. Behley, and C. Stachniss, "Poisson surface reconstruction for LiDAR odometry and mapping," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 5624–5630.
[11] J. Behley and C. Stachniss, "Efficient surfel-based SLAM using 3D laser range data in urban environments," in *Proc. Robot., Sci. Syst.*, vol. 2018, 2018, pp. 59–69.
[12] X. Chen, A. Milioto, E. Palazzolo, P. Giguere, J. Behley, and C. Stachniss, "SuMa++ : Efficient LiDAR-based semantic SLAM," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 4530–4537.
[13] A. Segal, D. Haehnel, and S. Thrun, "Generalized-ICP," *Proc. Robot., Sci. Syst.*, Seattle, USA, vol. 2, no. 4, pp. 435–443, 2009.
[14] M. Yokozuka, K. Koide, S. Oishi, and A. Banno, "LiTAMIN: LiDAR-based tracking and mapping by stabilized ICP for geometry approximation with normal distributions," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 5143–5150.
[15] J. W. Kim and B. H. Lee, "Robust and fast 3-D scan registration using normal distributions transform with supervoxel segmentation," *Robotica*, vol. 34, no. 7, pp. 1630–1658, 2016.
[16] T. Stoyanov, M. Magnusson, H. Andreasson, and A. J. Lilienthal, "Fast and accurate scan registration through minimization of the distance between compact 3D NDT representations," *Int. J. Robot. Res.*, vol. 31, no. 12, pp. 1377–1393, 2012.
[17] T. Stoyanov, M. Magnusson, and A. J. Lilienthal, "Point set registration through minimization of the L2 distance between 3D-NDT models," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2012, pp. 5196–5201.
[18] J. Zhang and S. Singh, "Visual-LiDAR odometry and mapping: Low-drift, robust, and fast," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2015, pp. 2174–2181.
[19] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "LIO-SAM: Tightly-coupled LiDAR inertial odometry via smoothing and mapping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 5135–5142.
[20] 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.*, 2018, pp. 4758–4765.
[21] Z. Zhao, W. Zhang, J. Gu, J. Yang, and K. Huang, "LiDAR mapping optimization based on lightweight semantic segmentation," *IEEE Trans. Intell. Veh.*, vol. 4, no. 3, pp. 353–362, Sep. 2019.
[22] S. W. Chen *et al.*, "SLOAM: Semantic LiDAR odometry and mapping for forest inventory," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 612–619, Apr. 2020.
[23] H. Lim, S. Hwang, S. Shin, and H. Myung, "Normal distributions transform is enough: Real-time 3D scan matching for pose correction of mobile robot under large odometry uncertainties," in *Proc. 20th Int. Conf. Control, Automat. Syst.*, 2020, pp. 1155–1161.
[24] M. Mielle, M. Magnusson, H. Andreasson, and A. J. Lilienthal, "SLAM auto-complete: Completing a robot map using an emergency map," in *Proc. IEEE Int. Symp. Saf., Secur. Rescue Robot.*, 2017, pp. 35–40.
[25] T. Stoyanov, J. Saarinen, H. Andreasson, and A. J. Lilienthal, "Normal distributions transform occupancy map fusion: Simultaneous mapping and tracking in large scale dynamic environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2013, pp. 4702–4708.
[26] J. Saarinen, H. Andreasson, T. Stoyanov, J. Ala-Luhtala, and A. J. Lilienthal, "Normal distributions transform occupancy maps: Application to large-scale online 3D mapping," in *Proc. IEEE Int. Conf. Robot. Automat.*, Karlsruhe, Germany, 2013, pp. 2233–2238.
[27] J. Lin and F. Zhang, "Loam livox: A fast, robust, high-precision LiDAR odometry and mapping package for LiDARs of small FoV," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 3126–3131.
[28] J. Lin and F. Zhang, "A fast, complete, point cloud based loop closure for LiDAR odometry and mapping," vol. abs/1909.11811, Sep. 2019, *arXiv:1909.11811*.
[29] M. Oelsch, M. Karimi, and E. Steinbach, "R-LOAM: Improving LiDAR odometry and mapping with point-to-mesh features of a known 3D reference object," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 2068–2075, Apr. 2021.
[30] C. Cadena *et al.*, "Past, present, and future of simultaneous localization and mapping: Towards the robust-perception age," *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1309–1332, Dec. 2016.
[31] G. Bresson, Z. Alsayed, L. Yu, and S. Glaser, "Simultaneous localization and mapping: A survey of current trends in autonomous driving," *IEEE Trans. Intell. Veh.*, vol. 2, no. 3, pp. 194–220, Sep. 2017.
[32] O. Agunbiade and T. Zuva, "Simultaneous localization and mapping in application to autonomous robot," in *Proc. Int. Conf. Intell. Innov. Comput. Appl.*, 2018, pp. 1–5.
[33] A. Zaganidis, M. Magnusson, T. Duckett, and G. Cielniak, "Semantic-assisted 3D normal distributions transform for scan registration in environments with limited structure," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 4064–4069.
[34] Z. Wang, J. Fang, X. Dai, H. Zhang, and L. Vlacic, "Intelligent vehicle self-localization based on double-layer features and multilayer LiDAR," *IEEE Trans. Intell. Veh.*, vol. 5, no. 4, pp. 616–625, Dec. 2020.
[35] M. Maaref, J. Khalife, and Z. M. Kassas, "Lane-level localization and mapping in GNSS-Challenged environments by fusing LiDAR data and cellular pseudoranges," *IEEE Trans. Intell. Veh.*, vol. 4, no. 1, pp. 73–89, Mar. 2019.
[36] J. Kümmerle, M. Sons, F. Poggenhans, T. Kühner, M. Lauer, and C. Stiller, "Accurate and efficient self-localization on roads using basic geometric primitives," in *Proc. Int. Conf. Robot. Automat.*, 2019, pp. 5965–5971.
[37] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation," in *Proc. Robot., Sci. Syst.*, 2015, pp. 1–10, doi: 10.15607/RSS.2015.XI.006.
[38] X. Chen, T. Läbe, A. Milioto, T. Röhling, J. Behley, and C. Stachniss, "OverlapNet: A siamese network for computing LiDAR scan similarity with applications to loop closing and localization," *Auton. Robots*, vol. 46, no. 1, pp. 61–81, 2022.
[39] G. Kim and A. Kim, "Scan context: Egocentric spatial descriptor for place recognition within 3D point cloud map," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 4802–4809.
[40] K. Koide, J. Miura, and E. Menegatti, "A portable three-dimensional LiDAR-based system for long-term and wide-area people behavior measurement," *Int. J. Adv. Robot. Syst.*, vol. 16, pp. 1–16, 2019.
[41] M. Quigley *et al.*, "ROS: An open-source robot operating system," in *Proc. IEEE Int. Conf. Robot. Automat. Workshop Open Source Robot.*, Kobe, Japan, 2009, pp. 5–11.
[42] 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.

Hongliang Guo received the B.E. degree in dynamic engineering and the M.E. degree in dynamic control from the Beijing Institute of Technology, Beijing, China, in 2005 and 2007, respectively, and the Ph.D. degree in electrical and computer engineering from the Stevens Institute of Technology, Hoboken, NJ, USA. From 2016 to 2020, he was an Associate Professor with the University of Electronics Science and Technology of China, Chengdu, China. In 2021, he joins the Institute of Infocomm Research, A*STAR, as a Scientist. His research interests include planning and learning under uncertainties.



Jiankang Zhu received the bachelor's degree from the School of Information Science and Engineering, Shenyang University of Technology, Shenyang, China, in 2019. He is currently a Graduate Student with the University of Electronic Science and Technology of China, Chengdu, China. His research interests include simultaneous localization and mapping, point cloud registration, and mobile robot related applications.

Yunping Chen (Member, IEEE) received the bachelor's and master's degrees from the Southwest University of China, Chongqing, China, and the Ph.D. degree in geographic information system and remote sensing from Beijing Normal University, Beijing, China. He is currently an Associate Professor with the University of Electronics Science and Technology of China, Chengdu, China. His research interests include remote sensing, geographic information system, mapping, and localization with multi-modal sensor information.

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on May 05,2024 at 06:13:56 UTC from IEEE Xplore. Restrictions apply.