Generalized LOAM: LiDAR Odometry Estimation with Trainable Local Geometric Features

Kohei Honda[1], Kenji Koide[2], Masashi Yokozuka[2], Shuji Oishi[2], and Atsuhiko Banno[2]

Abstract—This paper presents a LiDAR odometry estimation framework called Generalized LOAM. Our proposed method is generalized in that it can seamlessly fuse various local geometric shapes around points to improve the position estimation accuracy compared to the conventional LiDAR odometry and mapping (LOAM) method. To utilize continuous geometric features for LiDAR odometry estimation, we incorporate tiny neural networks into a generalized iterative closest point (GICP) algorithm. These neural networks improve the data association metric and the matching cost function using local geometric features. Experiments with the KITTI benchmark demonstrate that our proposed method reduces relative trajectory errors compared to the other LiDAR odometry estimation methods.

Index Terms—SLAM, Localization, Mapping, Deep Learning Methods, Computational Geometry.

I. INTRODUCTION

LIDAR odometry estimation with three-dimensional (3D) point clouds is essential for mobility technologies, such as autonomous mobile robots. To improve the accuracy and efficiency of the estimates, the use of surface geometric features estimated from point clouds is a promising approach. Some state-of-the-art LiDAR odometry estimation methods [1] employ feature-based registration derived from LiDAR odometry and mapping (LOAM) [2]. LOAM classifies points into edges and planes before registration. This approach enables efficient and accurate odometry estimates using local geometric shapes. However, LOAM has several downsides, such as the need for sensor-specific processing, difficulty in adjusting the balance between edges and planes, and degeneracy in environments where features cannot be well extracted.

Generalized iterative closest point (GICP) [3] is another popular method. GICP models the local surface at each point as a normal distribution and incorporates normal distribution comparison into the ICP matching cost. In the GICP framework, the sum of the distribution-to-distribution distances between corresponding points is minimized instead of the conventional point-to-point distance. Because a normal distribution can represent various local geometric shapes such as spheres, lines, or planes depending on the covariance matrix, GICP can be considered an ICP for a local-shape-to-local-shape comparison. The GICP cost function encompasses the edge and plane comparisons used in LOAM depending on the covariance matrices, and it can handle them seamlessly without explicitly separating the point cloud.

However, in the standard GICP, the normal distributions representing local geometric shapes around points are regularized as planes, which turns GICP into a plane-to-plane ICP [3]. This regularization is necessary to deal with rank deficient of covariance matrices, regardless of the density of the points or sensor noise. Unfortunately, this approximated representation leads to inaccurate representations of the local shapes and poorer registration performance. In addition, there is also room for improvement not only in the cost function, but also in the data association method, for example, in the metric-based ICP approach [4].

In this work, we propose a method called Generalized LOAM that fuses various local geometric shapes around points to improve the position estimation accuracy. Our proposed method is generalized in that it includes the line and plane comparison used in LOAM and fuses various surface shape features to assist in LiDAR odometry estimation. We reformulate a portion of the functions in the *classical* GICP method with neural networks to be able to seamlessly handle geometric features. First, we extract intermediate feature vectors using neural networks, such as PointNet++ [5] and RandLA-Net [6], which are trained for semantic segmentation. The feature vectors contain rich information on the local geometric shapes around points. Then, we improve the following two points in the GICP algorithm with intermediate feature vectors:

1) Reducing incorrect data associations by considering the features of points in addition to the Euclidean distance for nearest neighbor search.
2) Estimating reasonable covariance matrices for points with feature vectors. The covariance matrices represent local geometric shapes in the GICP matching cost. By improving the covariance estimation, we aim to make point comparisons robust in environments where plane-to-plane ICP estimation accuracy can deteriorate.

For each of data association and covariance estimation, we train tiny multi-layer perceptrons (MLPs) to transform the intermediate feature vectors of semantic segmentation networks into features dedicated to the specific tasks. Finally,

[1]Kohei Honda is with the Department of Mechanical Systems Engineering, Graduate School of Engineering, Nagoya University, Furo-cho, Chikusa-ku, Nagoya, Aichi, Japan, `honda.kohei.b0@s.mail.nagoya-u.ac.jp`

[2]Kenji Koide, Masashi Yokozuka, Shuji Oishi, and Atsuhiko Banno are with the Department of Information Technology and Human Factors, the National Institute of Advanced Industrial Science and Technology, Umezono 1-1-1, Tsukuba, 3050061, Ibaraki, Japan

through evaluation on the KITTI dataset [7], we demonstrated that the proposed framework improves the estimation accuracy compared to baselines.

In summary, the key contributions of this work are threefold:

1) We present a LiDAR odometry estimation framework called Generalized LOAM. This framework improves data association and covariance matrix estimation in the GICP method. We propose two tiny MLPs that convert local geometric feature vectors to suitable values for data association and matching cost evaluation with GICP.

2) We show that Generalized LOAM increases the accuracy of the estimated pose compared to the other existing methods. An ablation study confirmed the contribution of the two modified functions, data association, and covariance matrix estimation, to improve the estimation performance.

II. RELATED WORK

A. Improvement of ICP-based Algorithm Based on Features

Because our proposed method is an extension of the GICP algorithm [3], we describe improvements of the ICP-based method based on local features in detail.

There have been several studies on improving data association algorithms using local features. GF-ICP [8] improves the cost function for data association and rejects incorrect correspondences using the curvature, surface normal, and density of points. Gressin *et al.* used similar low-level local geometric features to find the optimal radius for the nearest neighbor search [9]. Color-GICP [10] finds the nearest neighbors in the extended dimensional space based on the L*a*b* color space.

Some studies proposed methods to estimate the normal distributions of points to better represent local geometric shapes. Surface-based GICP [11] estimates the surface normals for shapes using polynomial function fitting and weighted covariance matrices. To deal with the inhomogeneity of the point cloud, Mesh-GICP [12] estimates the direction of normal distributions from topological information with surface reconstruction. Multi-channel GICP [13] incorporates additional information, such as color and intensity, into the nearest neighbor search as kernel weights for estimating covariance matrices. These methods assume that all points belong to planes, which results in extracting low-dimensional local features.

B. Semantic Registration

Semantic registration methods extract geometric features, such as lines, planes, and spheres, from points, and efficiently determine the correspondences. LOAM [2] is the first method that divides point clouds into edge and plane points based on local curvature in a LiDAR scan line. While this method enables fast point cloud registration, its accuracy and robustness depend significantly on accurate point type labeling.

To achieve a more accurate and robust estimation, several modifications for semantic registration have been developed, including taking into account the uncertainty of the labels [14], [15], using multi-class semantic segmentation [16], [17], [18], [19], and adaptively changing the matching costs for

the planar and line points [20]. However, these approaches cannot take full advantage of the geometric features of a shape because they classify points into discrete labels and thus cannot seamlessly fuse the data into continuous geometric features.

C. Learning-based Odometry Estimation Approach

Several works have applied recent machine learning advances to odometry estimation. One type of method involves direct pose regression using end-to-end LiDAR odometry estimation neural networks [21], [22], [23]. These approaches do not require determining point-to-point correspondence, unlike the ICP-based method, and thus enable fast registration. However, their performance is typically limited to the dataset used for training, and their explainability is low.

Several other methods reformulate a subset of functions in the classical odometry estimation system with neural networks to achieve robust and explainable estimation [24], [25], [26]. These methods incorporate trainable features into a conventional optimization framework. Our proposed method also falls into this category.

III. GENERALIZED LOAM

A. System Overview

We aim to improve the following two points in the GICP algorithm with local geometric features:

1) Reducing incorrect point correspondences by considering local geometric features.
2) Providing reasonable point covariance matrices for robust point comparison in environments where plane-to-plane ICP estimation accuracy deteriorates.

Figure 1 shows the configuration of the proposed system. We first use a pre-trained neural network to estimate the local geometric features of points. We then reduce the dimensions of the features by principal component analysis (PCA) and transform them into representations dedicated to data association and covariance matrix regularization using tiny MLPs. Then, the pose is estimated following the GICP registration procedure.

B. GICP Registration

Because our proposed method is an extension of the GICP algorithm [3], we first introduce the details of this algorithm. We consider the estimation of a pose transformation T, which aligns a set of points $A = \{a_1, \ldots a_{N_A}\}$ (source point cloud) to another set of points $B = \{b_1, \ldots b_{N_B}\}$ (target point cloud). Following the conventional ICP scheme, GICP estimates the transformation between point clouds by repeating a two-step process: 1) estimating data association and 2) pose optimization by minimizing a cost function.

In the 1) data association step, the correspondences between source A and target B point clouds are estimated by the nearest neighbor search:

$$\{(a_i, b_i) \mid b_i = \underset{j \in \{1, \cdots, N_B\}}{\text{argmin}} \|\mathbf{p}_i^a - \mathbf{p}_i^b\|^2\}, \forall i \in \{1, \cdots, N_A\}, \quad (1)$$
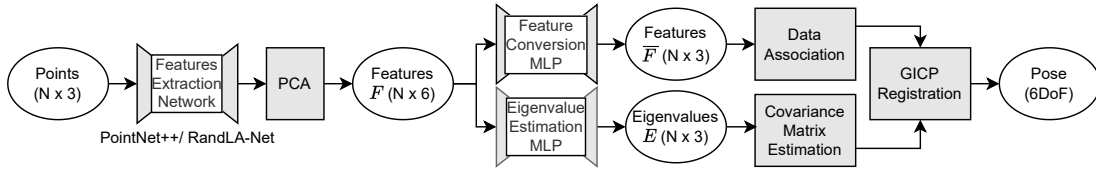
Fig. 1. Configuration of the proposed system. The proposed system estimates 6DoF poses on the GICP framework. We improve Data Association and Covariance Matrix Estimation using local geometric features extracted from the pre-trained backbone neural network.

where $\mathbf{p}_*$ are the Euclidean coordinates of points.

The GICP algorithm models the local surface around a point as a Gaussian distribution: $a_i \sim \mathcal{N}(\hat{a}_i, \boldsymbol{C}_i^A)$, $b_i \sim \mathcal{N}(\hat{b}_i, \boldsymbol{C}_i^B)$; $\boldsymbol{C}_i^A$ and $\boldsymbol{C}_i^B$ are the covariance matrices for points a_i and b_i, respectively. We assume that the transformation error becomes zero under the correct transformation; $\hat{d}_i = \hat{b}_i - \boldsymbol{T}\hat{a}_i = 0$. Then, the distribution of d_i is given by the reproductive property of the Gaussian distribution as

$$d_i \sim \mathcal{N}(\hat{b}_i - \boldsymbol{T}\hat{a}_i, \boldsymbol{C}_i^B + \boldsymbol{T}\boldsymbol{C}_i^A\boldsymbol{T}^T) = \mathcal{N}(0, \boldsymbol{C}_i^B + \boldsymbol{T}\boldsymbol{C}_i^A\boldsymbol{T}^T). \quad (2)$$

In the 2) transformation optimization step, the pose transformation $\hat{\boldsymbol{T}}$ is updated such that the log-likelihood of (2) is maximized:

$$\hat{\boldsymbol{T}} = \underset{\boldsymbol{T}}{\arg\max} \sum_i \log(p(d_i)) \quad (3)$$

$$= \underset{\boldsymbol{T}}{\arg\min} \sum_i d_i^T (\boldsymbol{C}_i^B + \boldsymbol{T}\boldsymbol{C}_i^A\boldsymbol{T}^T)^{-1} d_i. \quad (4)$$

This cost function is the sum of the Mahalanobis distances between corresponding points. It can be regarded as the weighting of the distances between the points by the covariance matrices $\boldsymbol{C}_i^A$ and $\boldsymbol{C}_i^B$. The covariance matrix for each point is typically estimated from its k neighbors and represents the local shape around the point.

For example, suppose the eigenvalues of the covariance matrix are $(a, \varepsilon, \varepsilon)$, (a, b, ε), and (a, b, c) where ε is a sufficiently small value. In this case, they represent a line, a plane, and a sphere, respectively. Thus, GICP can be considered as an ICP for local-shape-to-local-shape comparison.

However, in the original work on GICP [3], the covariance matrices are regularized by replacing their eigenvalues with $(1, 1, \varepsilon)$. This regularization turns GICP into a plane-to-plane ICP. This approximation can lead to incorrect or inadequate constraints, which cause pose estimation accuracy to deteriorate.

C. Estimation of Local Geometric Features

We estimate local geometric features using a pre-trained neural network for 3D point clouds. This work uses and compares Pointnet++ [5] and RandLA-Net [6], which are designed and trained for semantic segmentation. Because the outputs of these networks are the semantic labels of points, we expect their intermediate feature representations to contain rich information on the local geometric shapes around points. We extract the intermediate feature vectors from the 4th layer of PointNet++ and the 12th layer of RandLA-Net. The sizes of the PointNet++ and RandLA-Net feature vectors are $(N, 128)$ and $(N, 32)$, respectively, where N is the number of points.
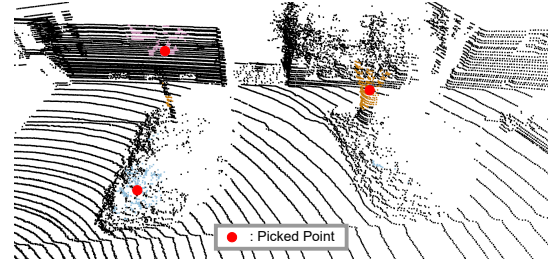
(a) Coloring of same semantic labels
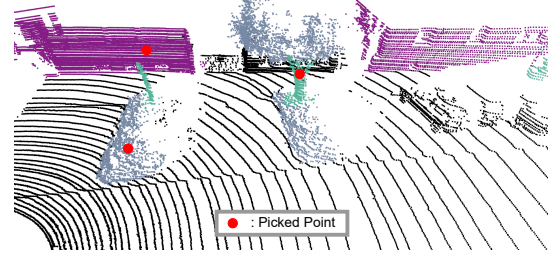
(b) Coloring of the close norm of feature vectors

Fig. 2. Semantic labels (a) and feature vectors (b) obtained from RandLA-Net [6]. For each picked point (red circles), we find the 500 closest neighboring points in the feature space and visualize them in the same color. In (b), the colored points surrounding the selected ones have the same labels or are similar in shape and are near in the distance. This result shows that the feature vectors contain information about local geometric shapes and spatial distances.

These features are used in a later process after they are compressed down to six-dimensional feature vectors $F = \{\mathbf{f}_1 \dots \mathbf{f}_N\}$ with PCA-based on singular value decomposition implemented in [27]. The reason for the compressioin is that the extracted raw feature vectors are sparse and the trained parameters need to be reduced, as described in section V. In our experiments on the KITTI dataset [7], the compressed features of PointNet++ and RandLA-Net have cumulative contribution rates of over 80% and 90%, respectively.

Figure 2 shows the semantic labels and feature vectors obtained from RandLA-Net. We find 500 nearest neighbor points in the feature space for each selected point (red circles) and visualize them in the same color. In Fig. 2, we can see that the colored points around the picked ones have the same labels or are similar in shape and are close in the distance. This result suggests that the feature vectors contain local geometric shapes and spatial distance information.

D. Data Association Using Local Geometric Features

Equation (1) in the standard GICP method states that the correspondences between source and target points are estimated by a nearest neighbor search based on the spatial distance of points. This metric is simple, and thus significant
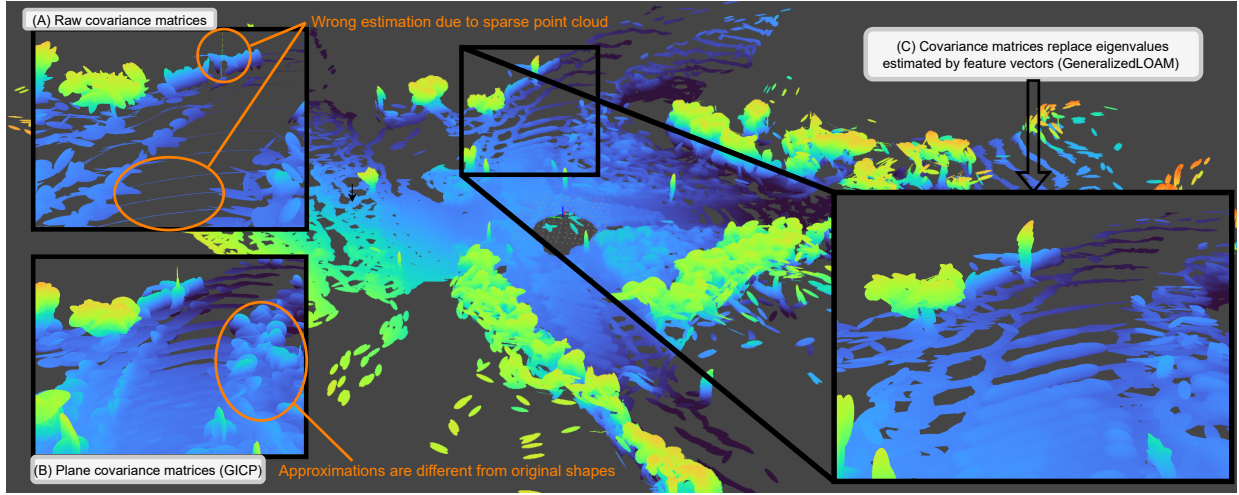
Fig. 3. Example of estimated covariance matrix shapes. (A) shows the raw covariance matrices estimated using only the positions of points. We can see that some points have degenerate distributions caused by the sparsity of pole and road surface points. The raw covariance matrices are regularized as planes in the standard GICP method. (B) shows the plane covariance matrices used in the standard GICP method. While this approach avoids degenerate distributions, the distributions of some points do not sufficiently represent the original surface shape. On the other hand, as shown in (C), the shapes of covariance matrices estimated by the proposed regularization algorithm better represent the surface shapes (planes, spheres, and lines) while avoiding the degeneracy of the distributions.

initial transformation errors and sensor noise can lead to incorrect associations and sub-optimal local solutions.

To reduce incorrect associations under transformation noise, we concatenate the local geometric features and the Euclidean coordinates of each point and find the nearest neighbors in the extended dimensional space. This is a similar approach to [10]. We use the L2 norm to measure the distance between feature vectors so that an efficient kd-tree algorithm [28] can be used for the nearest neighbor search:

$$\{(a_i, b_i) \mid b_i = \underset{j \in \{1, \cdots, N_B\}}{\operatorname{argmin}} (\|\mathbf{p}_i^a - \mathbf{p}_i^b\|^2 + \|\bar{\mathbf{f}}_i^a - \bar{\mathbf{f}}_i^b\|^2)\}, \quad (5)$$

$$\bar{\mathbf{f}}_i = \text{FeatureConversionMLP}(\mathbf{f}_i), \ \forall i \in \{1, \ldots N_A\}, \quad (6)$$

where $\mathbf{p}_*$ is the 3D coordinate of the point and $\mathbf{f}_* \in F$ is the estimated local geometric feature vector described in Section III-C. Note that positions and features are different physical quantities, and the 6-dimensional feature vector $\mathbf{f} \in F$ should not be used as-is. In [10], these scales are adjusted using weight parameters, but a trial-and-error adjustment is required. Instead of introducing weight parameters, we add a feature encoder (*Feature Conversion MLP*) before the data association, as shown in Fig. 1. This encoder is trained to convert the feature vector $\mathbf{f} \in F$ to an appropriate vector $\bar{\mathbf{f}} \in \bar{F}$ for the data association in Eq. (5). The encoder is a three-layer MLP consisting of six, four, and three nodes in the input, middle, and output layers, respectively. The ReLU activation function is used. The number of dimensions for output feature vector $\bar{\mathbf{f}} \in \bar{F}$ used in Eq. (5) is three. The training procedure for the *Feature Conversion MLP* will be described in section III-F.

E. Covariance Matrix Estimation Using Local Geometric Features

The covariance matrices in Eq. (4) represent the local shapes around points (line, plane, and sphere); their eigenvectors and eigenvalues respectively represent the basis vector directions

---

Algorithm 1 Covariance Matrix Estimation

Input: Point coordinates $\mathbf{p}_i \in P$, Point features $\mathbf{f}_i \in \bar{F}$, Number of neighbors k, Eigenvalue threshold ε

Output: Estimated covariance matrix $\boldsymbol{C}_i$

1: $M_i = \text{KNN}(k, \mathbf{p}_i, P)$ ▷ K-nearest neighbor search
2: $\mu_i = \frac{\sum_{\mathbf{p}_j \in M_i} \mathbf{p}_j}{k}$
3: $\boldsymbol{C}_i = \frac{\sum_{\mathbf{p}_j \in M_i} \mathbf{p}_j \mathbf{p}_j^T}{k} - \mu_i \mu_i^T$
4: $\boldsymbol{Q}_i \Lambda_i \boldsymbol{Q}_i^{-1} = \boldsymbol{C}_i$ ▷ Eigenvalue decomposition
5: $\mathbf{e}_i = \text{EigenvalueEstimationMLP}(\mathbf{f}_i)$
6: $\mathbf{e}_i' = (\text{Sort } \mathbf{e}_i \text{ elements in ascending order})$
7: **for** $e \in \mathbf{e}_i' = \{e_1, e_2, e_3\}$ **:**
8:      **if** $e \leq \varepsilon$ **:**
9:          $e = \varepsilon$ ▷ To prevent rank drop
10: $\mathbf{e}_i'' = \mathbf{e}_i'/\|\mathbf{e}_i'\|_2$ ▷ Scaling eigenvalues
11: $\boldsymbol{C}_i' = \boldsymbol{Q}_i \text{diag}(\mathbf{e}_i'') \boldsymbol{Q}_i^{-1}$ ▷ Replacing eigenvalues
12: **return** $\boldsymbol{C}_i'$

---

and lengths for point distributions. As described in section III-B, the standard GICP method replaces eigenvalues with $(1, 1, \varepsilon)$; All local shapes are approximated as planar with the same spread.

The approximation enables robust estimation of the geometric shapes regardless of the density of the points or sensor noise. However, the approximation can degrade the registration performance due to inaccurate representation of the local shapes.

Here, we propose a covariance regularization method with a tiny MLP to keep the expressive capability of covariance matrices while avoiding rank deficiency. Algorithm 1 describes the proposed covariance regularization method.

There are two key points in this algorithm:

1) Estimating principal component directions (i.e., eigenvectors) for a distribution based on the positions of the

nearest neighboring points.

2) Estimating eigenvalues that control the shape of the distribution with feature vectors F.

The eigenvalues are estimated by an encoder called *Eigenvalue Estimation MLP*, as shown in Fig. 1. This encoder is a three-layer MLP consisting of six, four, and three nodes in the input, middle, and output layers. The ReLU activation function is used to inject non-linearity.

Because the extracted local geometric features are trained to be rotation-invariant in Euclidean space [29] and do not contain directional information for the local shape, we estimate the eigenvectors with the standard PCA procedure and let the MLP estimate only the eigenvalues.

Figure 3 shows a visualization of an example of estimated covariance matrix shapes. Figure 3(A) shows the raw covariance matrices are estimated using only positions of points (line 3 in the Algorithm 1). We can see that some points have degenerate distributions caused by the sparsity of pole and road surface points. The raw covariance matrices are regularized as planes in the standard GICP method. Figure 3(B) shows the plane covariance matrices used in the standard GICP method. While this approach avoids degenerate distributions, the distributions of some points do not sufficiently represent the original surface shape. On the other hand, as shown in Fig. 3(C), the shapes of covariance matrices estimated by the proposed regularization algorithm better represent the surface shapes (not only planes but also spheres and lines) while avoiding degeneracy of the distributions.

F. Training Feature Conversion and Eigenvalue Estimation MLPs

We train the *Feature Conversion MLP* and *Eigenvalue Estimation MLP* using the KITTI dataset [7]. Figure 4 shows the pipeline for training the MLPs. Inspired by [24], we take a closed-loop training approach. We use the translational relative trajectory error (RTE) [31] between the estimated and ground-truth trajectories as the loss function $\mathcal{L}$:

$$\mathcal{L} = \frac{1}{N} \sum_s \sum_{s \in T} \|\delta \mathbf{t}_s^g - \delta \hat{\mathbf{t}}_s\|_2, \qquad (7)$$

where s represents a sub-trajectory with a fixed length cut from the trajectory T. $\delta \mathbf{t}_s^g$ and $\delta \hat{\mathbf{t}}_s$ are relative translational poses from the start point to the endpoint of each sub-trajectory, which are divided from the ground truth and estimated trajectories, respectively. N is the number sub-trajectory length settings and is used $N = 8$; (length of s) $\in \{100, 200, 300, 400, 500, 600, 700, 800\}$ [m] following the KITTI official evaluation code (*Development kit*).

Because the GICP registration module in the proposed system is non-differentiable, we apply a black-box optimization method based on the tree-structured Parzen estimator [32] for optimizing the weights of the MLPs. The implementation is performed in Optuna [33], and the number of weights for each MLP to be optimized is 46. The backbones of the *Features Extraction Network* (Pointnet++ and RandLA-Net) are pretrained on the SemanticKITTI dataset [30].

Figure 5 shows how the shapes of the estimated covariance matrix changed during the training process. Before the training (0 epochs), the MLPs with random weights represent a few planar shapes. As the training progresses from 10 epochs to 200 epochs, the estimated covariances start to better represent the surface as the loss decreases, but there is a large hole near the LiDAR. After training the MLPs for 500 epochs, the hole is filled, but the shapes of estimated covariance matrices tend to be "thorny" due to the sparsity of points and do not represent the actual local geometric shapes well. However, after 1000 epochs of training, they better represent local surface shapes while avoiding rank deficiency. The training changes the shapes significantly from the unlearning phase. This result suggests that minimization of the relative pose errors enables the MLP to represent local geometric shapes. In other words, it implies that the covariance matrices in the cost function (4) can improve the accuracy of the GICP by correctly approximating the local geometric shapes.

IV. EXPERIMENT

A. Implementation Details

We trained and evaluated the proposed Generalized LOAM method on the KITTI dataset. We trained the MLPs for 1000 epochs using five sequences (01, 03, 05, 07, 09) out of the 11 sequences of the KITTI dataset and used the rest as a validation set. Optimization of the GICP registration module was performed using the Levenberg-Marquardt optimizer in GTSAM[1].

B. Evaluation

We calculated the RTE averaged over 100 to 800 m trajectories with the KITTI official evaluation code (*Development kit*). We used an Intel Core i9-10900 (16 threads) to run the proposed framework.

1) Comparison with Baseline Methods: We compared the proposed method with four LiDAR odometry estimation algorithms: LOAM [2], LeGO-LOAM [1], plane-to-plane GICP [3], and SuMa [35]. We implemented both LOAM and GICP, and they share common processes (e.g., preprocessing and pose optimization) and parameters for a fair comparison. The LeGO-LOAM and SuMa results were obtained with parameters tuned by Optuna [33] in [34]. Note that all algorithms are compared using only the front-end part (loop closure is disabled). All quantitative evaluation results including rotational RTEs are available on a supplementary website [2].

Table I shows the translational RTEs. The proposed method using RandLA-Net gives the lowest RTEs (1.23% for the training set and 0.97% for the test set). The proposed method with PointNet++ presents the second-lowest and third-lowest RTEs (1.34% and 1.09%, respectively, for the training and test sets). This result suggests that the proposed method improves the accuracy of GICP due to high-quality local geometric feature representation. However, while the proposed method ranks first and second in the training dataset, SuMa gives

---

[1] https://github.com/borglab/gtsam
[2] See the project page for details: https://kohonda.github.io/proj-gloam/
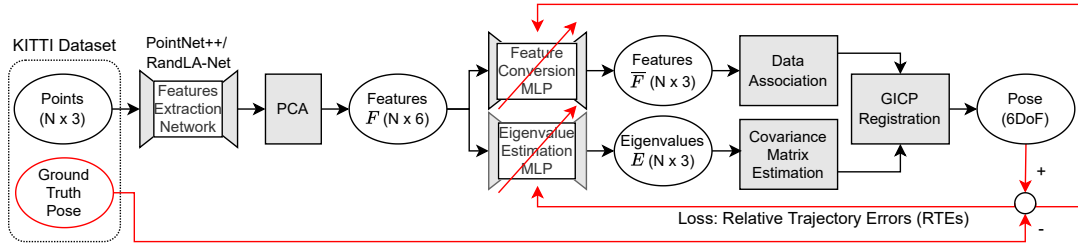
Fig. 4. Pipeline for training *Feature Conversion MLP* and *Eigenvalue Estimation MLP* on the KITTI dataset [7]. We use the loss function of the translational relative trajectory error (RTE) between the estimated and ground-truth trajectories. The backbones of the *Features Extraction Network* (Pointnet++ [5] and RandLA-Net [6]) are pre-trained on the SemanticKITTI dataset [30].
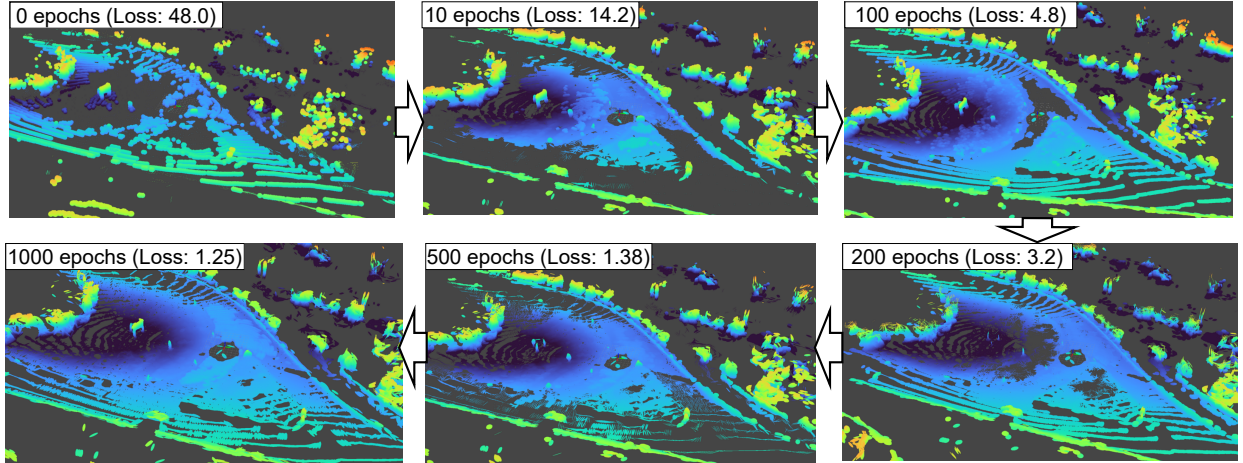

Fig. 5. Shapes of the estimated covariance matrix during the training process. In the early stages of training, the shapes of the estimated covariance matrices have many holes and thorns. As the learning progresses, they represent the actual local geometric shapes more closely as the loss decreases. This result suggests that the minimization of the relative pose errors enables the *Eigenvalue Estimation MLP* to represent local geometric shapes.

TABLE I
AVERAGE TRANSLATIONAL RTEs [%] FOR THE KITTI DATASET [7]

Seq. ID	Training dataset						Test dataset						
	01	03	05	07	09	Avg.	00	02	04	06	08	10	Avg.
LOAM	2.33	4.18	5.34	2.02	7.35	4.24	3.51	10.8	3.40	1.19	4.66	4.08	4.61
LeGO-LOAM	3.07	1.63	1.02	1.02	1.29	1.61	1.87	1.83	1.33	1.09	1.76	1.83	1.62
GICP	3.18	1.06	0.90	0.57	1.26	1.40	1.11	1.54	1.07	0.72	1.15	1.40	1.17
SuMa	5.59	1.14	0.87	0.66	0.86	1.82	0.84	1.27	0.56	0.60	1.43	1.73	1.07
Generalized LOAM (PN++)	3.16	1.04	0.82	0.53	1.15	1.34	1.04	1.45	1.01	0.68	1.10	1.24	1.09
Generalized LOAM (RN)	2.91	0.95	0.76	0.53	1.00	1.23	0.94	1.33	0.87	0.64	1.06	0.99	0.97

Red and blue respectively indicate the first and second best results. PN++ and RN mean PointNet++ and RandLA-Net, respectively. LOAM and GICP are implemented by ourselves. LeGO-LOAM and SuMa show the results with parameters tuned by Optuna in [34].

the second-lowest average and lowest RTEs in Seqs. 00, 02, 04, and 06 of the test dataset. This result indicates that the proposed methods are slightly over-trained. We expect that the generalization performance can be improved by training on various data other than the KITTI dataset and enhancing the data efficiency of training described in section V.

2) Processing Time: We compared the processing times for each process for all the methods, and the results are shown in Fig. 6. Note that the computation time of LOAM is only for the front-end part. Because feature extraction using PointNet++ takes an average of 261 ms and is not suitable for real-time execution, only the proposed method using RandLA-Net is shown in Fig. 6. The proposed method using RandLA-Net takes roughly 30 ms for processing a frame with multi-threading and thus is sufficiently faster than the real-time requirement (100 ms per frame). The processing times are,

however, slightly increased compared to the baseline methods. This is because the number of dimensions is increased from 3 to 6, which affects the efficiency of the nearest neighbor search for data association.

3) Ablation Study: Our proposed method (Generalized LOAM) improves two functions of GICP using local geometric features; data association and covariance matrix estimation. We compared the RTEs for GICP with four settings: the conventional GICP, GICP with the proposed data association, GICP with the proposed covariance matrix estimation, and Generalized LOAM to confirm that both functions contribute to the improvement in accuracy. We show the results of these ablation studies using the test dataset (Seq. (00, 02, 04, 06, 08, 10)) in Table II. GICP with two functions (=Generalized LOAM) shows the best result with Pointnet++ and RandLA-Net for all the sequences. GICP with either function improves

TABLE II
ABLATION STUDY ON KITTI DATASET [7]

Method/Seq.		00		02		04		06		08		10		mean	
		t_{RTE}[1]	r_{RTE}[1]	t_{RTE}	r_{RTE}	t_{RTE}	r_{RTE}	t_{RTE}	r_{RTE}	t_{RTE}	r_{RTE}	t_{RTE}	r_{RTE}	t_{RTE}	r_{RTE}
GICP (Baseline)		1.11	0.52	1.54	0.55	1.07	0.67	0.72	0.36	1.15	0.47	1.40	0.61	1.27	0.52
GICP+DA [2]	PN++ [3]	1.06	0.50	1.48	0.53	1.03	0.64	0.70	**0.35**	1.14	0.45	1.32	0.57	1.24	0.50
	RN [3]	1.07	0.50	1.49	0.53	1.02	0.64	0.68	0.35	1.12	0.45	1.28	0.55	1.22	0.50
GICP+CE [2]	PN++	1.08	0.51	1.52	0.54	1.06	0.65	0.70	**0.35**	1.12	0.45	1.31	0.58	1.24	0.51
	RN	1.00	0.47	1.40	0.48	0.93	0.57	0.68	0.34	1.08	0.42	1.12	0.51	1.15	0.46
GICP+DA+CE	PN++	**1.04**	**0.49**	**1.45**	**0.52**	**1.01**	0.62	0.68	0.35	**1.10**	0.43	**1.24**	0.54	**1.20**	**0.48**
(Generalized LOAM)	RN	**0.94**	**0.43**	**1.33**	**0.46**	**0.87**	**0.53**	**0.64**	**0.33**	**1.06**	**0.40**	**0.99**	**0.45**	**1.09**	**0.43**

[1] t_{RTE} and r_{RTE} are the translation [%] and rotation [°/m] RTEs.
[2] DA and CE mean data association and covariance matrix estimation using features, which are described in Section III-D and III-E, respectively.
[3] PN++ and RN mean PointNet++ and RandLA-Net, respectively.
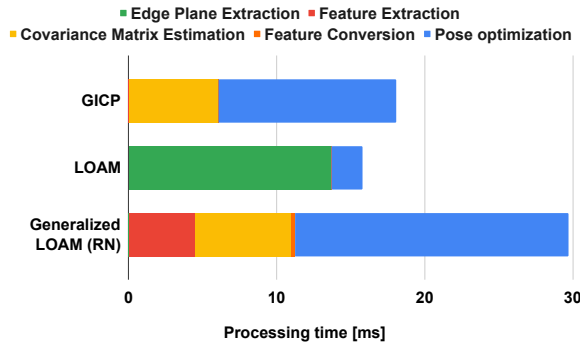

Fig. 6. Average processing time per scan. Processing is parallelized with 16 threads. RN means RandLA-Net. Generalized LOAM takes roughly 30 ms to process a frame with multi-threading and thus is sufficiently faster than the real-time requirement (100 ms per frame).

the RTEs compared to the baseline GICP (PN++: -0.07%, $-0.04°$/m; RN: -0.18%, $-0.09°$/m). For RandLA-Net, the improvement of the mean RTEs with covariance matrix estimation (-0.12% and $-0.06°$/m) is more significant than that with data association (-0.05% and $-0.02°$/m). This could be the reason that the backbones (PontNet++ and RandLA-Net) are trained for the semantic segmentation task and that the extracted features are better suited for approximating local geometric shapes than data association. We expect that the data association could be further improved by re-training the backbone with data association.

C. Real Data Experiment

The Generalized LOAM illustrated in Fig.1 uses local geometric features extracted from PointNet++ or RandLA-Net as a backbone. Because the two neural networks are pre-trained with the KITTI dataset, the technical characteristics of the LiDAR model (Velodyne HDL-64E[3]) may influence the features. To verify the generalization performance of the proposed method, we tested our system on another dataset acquired with a different LiDAR model.

We measured point cloud data (4142 frames) in an outdoor environment using an Ouster OS0-64 model[4]. Although the number of scan lines in the OS0-64 model is the same as that in Velodyne HDL-64E, the scan angle and range differ.

[3] https://velodynelidar.com/products/
[4] https://ouster.com/jp/products/scanning-lidar/os0-sensor/

Figure 7 shows a point cloud map and estimated covariance matrices generated by Generalized LOAM with RandLA-Net. This result shows that the proposed system works well without significant corruption and estimates high-quality local geometric shapes as covariance matrices.

We also compared the pose estimation errors at the start and end frames with the plane-to-plane GICP. The error was 2.56m / 0.023rad for GICP and 1.86m / 0.007rad for Generalized LOAM using RandLA-Net. This result suggests that the proposed method will perform well when generalized to other LiDAR models.

V. CONCLUSION AND FUTURE WORK

This paper presents a point cloud registration method that improves data association and covariance matrix estimation in the GICP method using local geometric features. We propose two MLPs: the *feature conversion MLP* converts local geometric features to scaled values for the metric for nearest neighbor search, and the *eigenvalues estimation MLP* estimates eigenvalues of the covariance matrices to avoid rank deficient covariance matrices while keeping their expressive capability. These two MLPs are trained with closed-loop pose estimation and an error evaluation pipeline.

In the current implementation, the proposed system involves non-differentiable processes that limit the number of trainable parameters in the system because efficient differentiation-based optimization is not applicable. Additionally, although the current loss function in Eq. (7) is a typical and direct evaluation metric for odometry estimation, its results are too sparse to provide teaching data for training; we can obtain only one error for each continuous sequence on the KITTI dataset, which is inefficient for training. In future work, we plan to further improve registration accuracy by making the entire process differentiable and improving the loss function.

REFERENCES

[1] T. Shan and B. Englot, "Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain," in International Conference on Intelligent Robots and Systems, 2018, pp. 4758–4765.
[2] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time," in Robotics: Science and Systems, vol. 2, no. 9, 2014, pp. 1–9.
[3] A. Segal, D. Haehnel, and S. Thrun, "Generalized-icp." in Robotics: science and systems, vol. 2, no. 4, 2009, p. 435.
[4] L. Armesto, J. Minguez, and L. Montesano, "A generalization of the metric-based iterative closest point technique for 3d scan matching," in International Conference on Robotics and Automation, 2010, pp. 1367–1372.
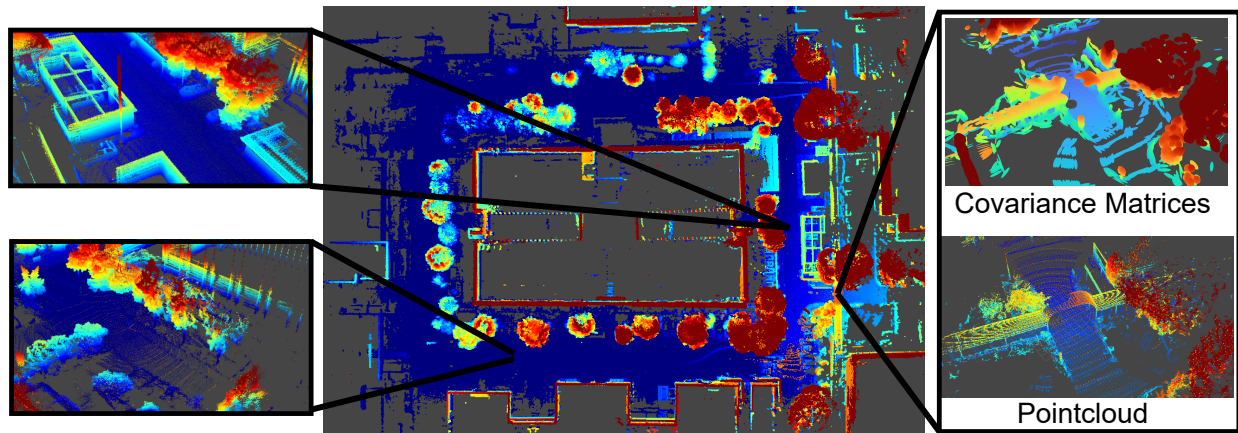
Fig. 7. Estimation result on a dataset acquired with Ouster OS1-64. Our proposed system works without major breakdowns and estimates the high-quality local geometric shapes as covariance matrices with a LiDAR model that is different from the one used for the acquisition of the training dataset.

[5] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," Advances in Neural Information Processing Systems, vol. 30, 2017.

[6] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham, "Randla-net: Efficient semantic segmentation of large-scale point clouds," in Conference on Computer Vision and Pattern Recognition, 2020, pp. 11 108–11 117.

[7] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in Conference on Computer Vision and Pattern Recognition, 2012, pp. 3354–3361.

[8] Y. He, B. Liang, J. Yang, S. Li, and J. He, "An iterative closest points algorithm for registration of 3d laser scanner point clouds with geometric features," Sensors, vol. 17, no. 8, p. 1862, 2017.

[9] A. Gressin, C. Mallet, J. Demantké, and N. David, "Towards 3d lidar point cloud registration improvement using optimal neighborhood knowledge," Journal of Photogrammetry and Remote Sensing, vol. 79, pp. 240–251, 2013.

[10] M. Korn, M. Holzkothen, and J. Pauli, "Color supported generalized-icp," in International Conference on Computer Vision Theory and Applications, vol. 3, 2014, pp. 592–599.

[11] M. Vlaminck, H. Luong, and W. Philips, "Surface-based gicp," in Conference on Computer and Robot Vision, 2018, pp. 262–268.

[12] D. Holz and S. Behnke, "Registration of non-uniform density 3d point clouds using approximate surface reconstruction," in ISR/Robotik International Symposium on Robotics. VDE, 2014, pp. 1–7.

[13] J. Servos and S. L. Waslander, "Multi channel generalized-icp," in International Conference on Robotics and Automation, 2014, pp. 3644–3649.

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

[15] N. Akai, T. Hirayama, and H. Murase, "Semantic localization considering uncertainty of object recognition," Robotics and Automation Letters, vol. 5, no. 3, pp. 4384–4391, 2020.

[16] X. Chen, A. Milioto, E. Palazzolo, P. Giguere, J. Behley, and C. Stachniss, "Suma++: Efficient lidar-based semantic slam," in International Conference on Intelligent Robots and Systems, 2019, pp. 4530–4537.

[17] Y. Pan, P. Xiao, Y. He, Z. Shao, and Z. Li, "Mulls: Versatile lidar slam via multi-metric linear least square," in International Conference on Robotics and Automation, 2021, pp. 11 633–11 640.

[18] A. Zaganidis, L. Sun, T. Duckett, and G. Cielniak, "Integrating deep semantic segmentation into 3-d point cloud registration," Robotics and Automation Letters, vol. 3, no. 4, pp. 2942–2949, 2018.

[19] F. Wang, Z. Wang, F. Yan, H. Gu, and Y. Zhuang, "A novel real-time semantic-assisted lidar odometry and mapping system," in Tenth International Conference on Intelligent Control and Information Processing, 2019, pp. 44–50.

[20] Q. Sun, J. Yuan, X. Zhang, and F. Duan, "Plane-edge-slam: Seamless fusion of planes and edges for slam in indoor environments," Transactions on Automation Science and Engineering, vol. 18, no. 4, pp. 2061–2075, 2020.

[21] M. Velas, M. Spanel, M. Hradis, and A. Herout, "Cnn for imu assisted odometry estimation using velodyne lidar," in International Conference on Autonomous Robot Systems and Competitions, 2018, pp. 71–77.

[22] Q. Li, S. Chen, C. Wang, X. Li, C. Wen, M. Cheng, and J. Li, "Lo-net: Deep real-time lidar odometry," in Conference on Computer Vision and Pattern Recognition, 2019, pp. 8473–8482.

[23] Y. Cho, G. Kim, and A. Kim, "Unsupervised geometry-aware deep lidar odometry," in International Conference on Robotics and Automation, 2020, pp. 2145–2152.

[24] P.-E. Sarlin, A. Unagar, M. Larsson, H. Germain, C. Toft, V. Larsson, M. Pollefeys, V. Lepetit, L. Hammarstrand, F. Kahl, et al., "Back to the feature: Learning robust camera localization from pixels to pose," in Conference on Computer Vision and Pattern Recognition, 2021, pp. 3247–3257.

[25] J. K. Murthy, S. Saryazdi, G. Iyer, and L. Paull, "gradslam: Dense slam meets automatic differentiation," in International Conference on Robotics and Automation, 2020.

[26] C. Tang and P. Tan, "BA-net: Dense bundle adjustment networks," in International Conference on Learning Representations, 2019.

[27] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.

[28] J. L. Bentley, "Multidimensional binary search trees used for associative searching," Communications of the ACM, vol. 18, no. 9, pp. 509–517, 1975.

[29] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in Conference on Computer Vision and Pattern Recognition, 2017, pp. 652–660.

[30] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, "SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences," in International Conference on Computer Vision, 2019, pp. 9297–9307.

[31] Z. Zhang and D. Scaramuzza, "A tutorial on quantitative trajectory evaluation for visual (-inertial) odometry," in International Conference on Intelligent Robots and Systems, 2018, pp. 7244–7251.

[32] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyperparameter optimization," Advances in Neural Information Processing Systems, vol. 24, 2011.

[33] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in International Conference on Knowledge Discovery and Data Mining, 2019.

[34] K. Koide, M. Yokozuka, S. Oishi, and A. Banno, "Adaptive hyperparameter tuning for black-box lidar odometry," in International Conference on Intelligent Robots and Systems. IEEE, 2021, pp. 7708–7714.

[35] J. Behley and C. Stachniss, "Efficient surfel-based slam using 3d laser range data in urban environments," in Robotics: Science and Systems, vol. 2018, 2018, p. 59.