Global Localization in Large-scale Point Clouds via Roll-pitch-yaw Invariant Place Recognition and Low-overlap Global Registration

Zhong Wang, Lin Zhang, *Senior Member, IEEE,* Shengjie Zhao, *Senior Member, IEEE*, and Yicong Zhou, *Senior Member, IEEE*

***Abstract*—On platforms like autonomous ground vehicles, global localization with 3D LiDAR is an indispensable part of tasks such as navigation or simultaneous localization and mapping. Usually, LiDAR global localization is subdivided into two sub-problems, place recognition and global registration. For place recognition, the recent emerging schemes based on deep learning either rely on 3D convolution with high complexity or need to learn features from varying forward perspectives. To mitigate this, we propose RpyPR, which represents point clouds as probabilistic voxels and generates occupancy grids from a bird's-eye view, fulfilling robust place recognition by learning aggregated embeddings from a fixed perspective. For low-overlap global registration, the traditional handcraft-based methods are mostly limited to dense object-level point clouds, while the state-of-the-art learning-based approaches often rely on complex 3D convolution and additional feature association learning. To fill this gap to some extent, we propose LoPcGR which estimates the relative roll-pitch-Δz parameters by fitting and aligning the ground plane of the point clouds and determines Δx-Δy-yaw values by matching their projected occupancy grids. Extensive experiments corroborate RpyPR's superior recall and generalization ability, as well as LoPcGR's advanced success rate and accuracy. Especially in the recognition and registration of hard samples, our results far exceed those of our counterparts by large margins. To ensure full reproducibility, the relevant codes and data are made available online[1].**

***Index Terms*—Global Localization, Place Recognition, Low-overlap Global Registration, Point Cloud.**

I. Introduction

AFTER years of development, intelligent agents have gradually entered our public life, such as mobile robots and self-driving cars. For such unmanned ground vehicles, to achieve their friendly interaction with the surrounding environments, a prerequisite is to accurately estimate their global poses in the existing environmental maps, which is called a global localization problem (or re-localization in simultaneous localization and mapping). In GPS-denied environments, LiDAR (Light Detection and Ranging) has become one of the mainstream sensors relied on to fulfill such a goal, owing to its merits of high accuracy ranging, wide-angle viewing, and active measuring [1].

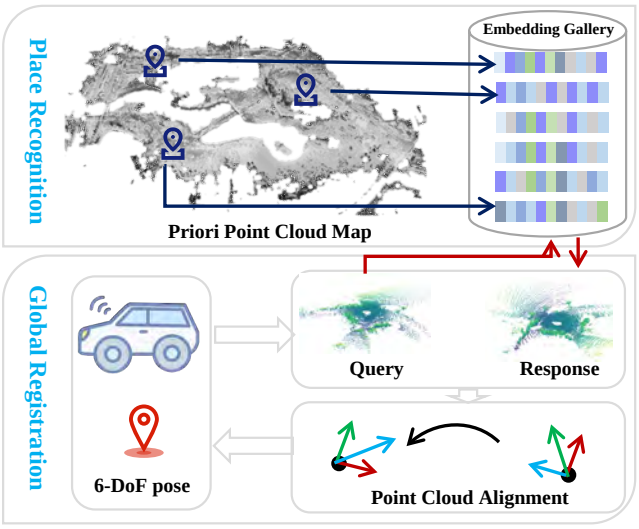


Fig. 1: The general architecture of a typical global localization system.

Using a divide-and-conquer strategy, LiDAR global localization can be divided into two sub-problems, i.e., place recognition and global registration [2]. As presented in Figure 1, the former is responsible for retrieving response candidates with high similarities from the historical gallery using the query point cloud, while the latter is devoted to accurately aligning the query point cloud to the response one and thus obtaining its pose in the global frame.

To date, LiDAR place recognition mainly follows the technical route of visual loop closure detection [3]–[5], i.e., by constructing a gallery of place embeddings from historical point clouds and detecting similar places resorting to fast retrieval techniques. According to the generation of place embeddings, existing LiDAR place recognition methods can be classified into two categories: handcraft-based and learning-based. The handcraft-based ones focus on the construction of global place descriptors with yaw invariances, such as histogram-based [6]–[9], matrix factorization-based [10], grid-based [11]–[13], and frequency-based [14], [15]. The major

This work was supported in part by the National Key Research and Development Project under Grant 2020YFB2103900; in part by the National Natural Science Foundation of China under Grant 62272343 and Grant 61973235; in part by the Shanghai Science and Technology Innovation Plan under Grant 20510760400; in part by the Shuguang Program of Shanghai Education Development Foundation and Shanghai Municipal Education Commission under Grant 21SG23; and in part by the Fundamental Research Funds for the Central Universities. (*Corresponding author: Lin Zhang.*)

Zhong Wang, Lin Zhang, and Shengjie Zhao are with the School of Software Engineering, Tongji University, Shanghai 201804, China, and also with the Engineering Research Center of Key Software Technologies for Smart City Perception and Planning, Ministry of Education (email: {2010194, cslinzhang, shengjiezhao}@tongji.edu.cn).

Yicong Zhou is with the Department of Computer and Information Science, University of Macau, Macau 999078, China (e-mail: yicongzhou@um.edu.mo).

[1]https://cslinzhang.github.io/GLoc/GLoc.html

advantages of these methods are that they are computationally efficient, interpretable, and training-free. However, they often require adjustment of many hyperparameters, therefore leading to low robustness in practical applications. Another class of learning-based schemes can be finer categorized into two types. One is to learn place embeddings directly from the raw point cloud end-to-end. For example, PointNetVLAD [16] and LPD-Net [17] extract features for each point based on PointNet [18], and the aggregated embedding of these features will be learned subsequently with Net Vector of Locally Aggregated Descriptors (NetVLAD) [19]. Such neural networks can be trained end-to-end, but they often require downsampling the raw point cloud dozens of times due to the high computational cost of processing the entire scan, which inevitably undermines their performance. The other class first converts point clouds into pseudo-camera images (e.g., range images or semantic images [20]–[22], etc.) and subsequently learns place embeddings from them using image-oriented deep learning. These methods take advantage of the fact that 2D convolution can be conducted efficiently. However, on the one hand, the scenes in the forward perspective are often complex and variable, thus making it challenging to train robust place embeddings directly from those images generated from the pseudo-camera view. On the other hand, they tend to consider only the rotational invariance in yaw and lack the rotational invariances in roll and pitch, which deteriorates their performance when query point clouds have great differences in poses with the ones in the historical gallery.

For point cloud registration, traditional methods such as Iterative Closest Point (ICP) [23], Normal Distributions Transform (NDT) [24], and their variants [25]–[27] can achieve satisfactory results when the approximate initial values are available. However, they are limited to only local registration methods. That is to say, when lacking approximate initial guesses, these methods only produce local optimal solutions. In the case of global localization, there is often a difference of several meters between the query and the response frame, and also there may be great attitude differences in roll, pitch, and yaw. Such a challenging registration with a low overlap between two point clouds is also called a global registration problem [28], [29]. To solve this problem, a few methods based on parameter search estimate the global pose by branching and bounding the solution space step by step [30], [31]. They perform well in object-level registration between dense point clouds, but when encountering registrations of large-scale sparse point clouds like LiDAR scans, their efficiency and results will be greatly reduced. Another line of approaches establishes the corresponding relationship between the query and response points by minimizing the metric feature distance of their salient points, thus transforming the pose estimation into a least squares problem. Among them, the ones based on handcraft features are mostly limited to object-level registration [32], [33]. Although the learning-based ones yield state-of-the-art results recently, they often require complex 3D convolution and additional feature association learning to eliminate outliers, which undoubtedly increases the computational complexity [28], [29], [34]–[36].

In this article, to fill the aforementioned research gap to a certain extent, we propose a **R**oll-**p**itch-**y**aw invariant **P**lace **R**ecognition scheme, termed **RpyPR**, and a **L**ow-**o**verlap **P**oint **c**loud **G**lobal **R**egistration approach, **LoPcGR** in short. Specifically, considering that the variable scene from the forward perspective makes it difficult to establish place embeddings with significant discrimination, we choose to generate the occupancy grids from point clouds in the bird's-eye view and learn aggregated features end-to-end from these grids. With such a representation, each element of an occupancy grid contains the sum of the probabilities that the position is occupied by a point cloud in its vertical direction. In this way, on the one hand, the occupancy grid of the same place will be roll and pitch invariant. On the other hand, the distribution of features in the same place will be more concentrated in the feature space, which brings higher distinguishability of different places. After retrieving similar candidates from the gallery, we conduct the 6 degrees of freedom (DoF) registration of the query frame and the response frame in two steps, that is, first determine the roll angle (α), pitch angle (β), and vertical translation (Δz), and estimate the horizontal translations (Δx and Δy) and yaw angle (γ) afterwards. Specifically, since the working scenes of mobile robots and unmanned ground vehicles generally have an approximately horizontal ground, that shared ground plane is therefore regarded as a static reference. Firstly, we extract the ground points from the point cloud and perform plane fitting, so as to align the LiDAR coordinate system with the ground's frame to estimate α, β, and Δz. Secondly, feature points of the projected occupancy grids are extracted and matched, and the 2D transformation among them is estimated, thus obtaining reasonable estimates of Δx, Δy, and γ. As a result, the full 6-DoF pose can be restored from α, β, γ, Δx, Δy, and Δz.

Extensive experiments are conducted on large-scale outdoor datasets KITTI [37], [38] and NCLT [39] to evaluate the performance of the proposed RpyPR and LoPcGR. The results show that our RpyPR outperforms its counterparts by a large margin on KITTI, with a top-1 recall rate 7.5 percentage points higher than that of the runner-up. The evaluations on distinct scenarios also demonstrate the superior generalization ability of our RpyPR. More specifically, RpyPR trained on KITTI achieves a top-1 recall rate of 98.92% on NCLT and that trained on NCLT obtains a top-1 recall rate of 85.40% on KITTI. In global registration between low-overlap point clouds, our LoPcGR achieves the highest success rates at all difficulty levels, especially far surpassing the state-of-the-art approaches in medium and hard registrations. Additionally, LoPcGR also achieves pleasing registration accuracy, with translation and rotation errors of 0.20m and 0.26°. Last but not least, the practicability of RpyPR and LoPcGR is also corroborated in real-world applications both of global localization and SLAM.

To summarize, the characteristics of our approaches and our contributions are threefold:

1) We propose **RpyPR** for place recognition in large-scale point clouds of unmanned ground vehicles. RpyPR transforms point clouds into occupancy grids and learns place embeddings from the bird's-eye view. Such a probabilistic representation markedly reduces the morpho-

logical differences of point clouds in different attitudes, bringing superior generalization ability to RpyPR and making it roll-pitch-yaw invariant.

2) We propose **LoPcGR**, a global registration approach for 6-DoF matching of low-overlap point clouds. By aligning point clouds to the ground and matching their occupancy grids, LoPcGR separately estimates roll-pitch-Δz and Δx-Δy-yaw parameters in two steps. Such a strategy makes it achieve high success rates and accuracy even when there are large pose differences between the point clouds to be registered.
3) We demonstrate the practicability of the proposed RpyPR and LoPcGR in global localization and SLAM and verify their great recall, high success rates, and precise registration through extensive experiments. Also, to ensure the full reproducibility of our work and for the convenience of the community, all the relevant codes and data are made online available[1].

II. Related Work

A. LiDAR Place Recognition

According to the construction manners of place embeddings, the existing LiDAR place recognition methods can be roughly classified into handcraft-based ones and learning-based ones. The handcraft-based schemes focus on the design of place descriptors with statistical characteristics, including histogram-based ones [6]–[8], matrix factorization-based ones [10], grid-based ones [11]–[13], and frequency-based ones [14], [15]. For example, VFH [6], CVFH [7], and Small-sized Signatures [8] calculate the normal vector of the point cloud and encode the histogram of the relative geometric angles as the place embedding. In a matrix factorization way, M2DP [10] projects the point cloud onto multiple 2D planes from different perspectives and subsequently conducts Singular Value Decomposition on the obtained projection matrices. Afterwards, the singular vectors of those matrices constitute a 192-dimensional compact representation of the measurement place. Compared with those histogram-based methods, M2DP improves recall by a large margin at the expense of losing some computational efficiency. In a grid-representation manner, Kim *et al.* proposed Scan-Context [12], which divides the point cloud into fan-shaped grids according to the polar coordinates of the points and generates a featured image of the place from those grids according to the maximum z values of the points belonging to the grids. To further speed up the retrieval, Kim *et al.* extracted the seminal pixels from the feature image row by row and constructed the yaw-invariant ring key. Turning eyes to the frequency domain, Wang *et al.* put forward LiDAR-Iris [14], which transforms point clouds into frequency images and generates binary signatures corresponding to each point cloud by LoG-Gabor filtering and thresholding. When searching, the revisited place is detected by comparing the Hamming distance between those signatures. Recently, Cui *et al.* proposed Bow3D [9], which constructs a bag-of-words model of place embeddings based on Link3D [40], actively exploring the technology migration of visual loop detection to LiDAR place recognition.

With the rapid development of deep learning, recent years have witnessed a growing interest in learning-based LiDAR place recognition. Among those learning-based approaches, PointNetVLAD [16], as a distinguished pioneer, extracts the feature vector of each point based on the popular point cloud encoder PointNet [18] and learns the aggregated place embedding using a NetVLAD layer [19]. To make up for the deficiency of PointNetVLAD in modeling geometric relationships, LPD-Net [17] designs an Adaptive Local Feature Extraction module to construct several handcraft features of each point. After that, these handcraft features, together with the raw points, are fed into a PointNet-like network to extract the point-wise features. These features are further encoded via an attention network, and global descriptors are aggregated eventually by PointNet and NetVLAD again. Beyond the place recognition as PointNetVLAD and LPD-Net do, Du *et al.* proposed DH3D [34] to fulfill large-scale 6-DoF relocalization, which is based on a siamese structure, employs FlexConv and Squeeze-and-Excitation to capture multi-scale geometric information along with channel-wise association and jointly infers local and global features. Different from the above methods of learning features directly from 3D point clouds, some schemes first generate 2D images from point clouds and then extract place embeddings resorting to convolutional neural networks. For example, OverlapNet [20] generates range, normal, intensity, and semantic images in forward view and employs a siamese network to predict the overlap between two frames of point clouds and regress the relative rotation in yaw at the same time. In [21], Ma *et al.* introduced the Transformer [41] structure into OverlapNet [20], yielding OverlapTransformer, which further boosts the performance of place recognition compared with OverlapNet.

B. Point Cloud Registration

As analyzed in Section I, according to whether the initial values are provided, existing point cloud registration methods can be divided into two categories: local registration and global registration. One seminal approach of local point cloud registration is Iterative Closest Point (ICP) [23], which minimizes the geometric distance between two point sets, alternately optimizes rotation and translation, and iteratively seeks the optimal solution. In order to be more robust against outliers, Biber *et al.* proposed Normal Distribution Transform (NDT) [24], which casts the registration problem as a parameter estimation problem for similar distributions. Although both ICP and NDT can obtain satisfactory results when reasonable initial values are available, when such a condition is not met, what they produce is often a locally optimal solution. To find the global optimum, some researchers were devoted to designing fast search methods in the solution space. For example, Olsson *et al.*'s approach [42] and Go-ICP [31] are based on branch-and-bound that recursively compresses the parameter search space by finding the upper bound of the optimal matching. Another class of schemes focuses on establishing salient points together with descriptors and establishing point correspondence by minimizing the metric distance between the descriptors to estimate the alignment parameters. In such approaches,

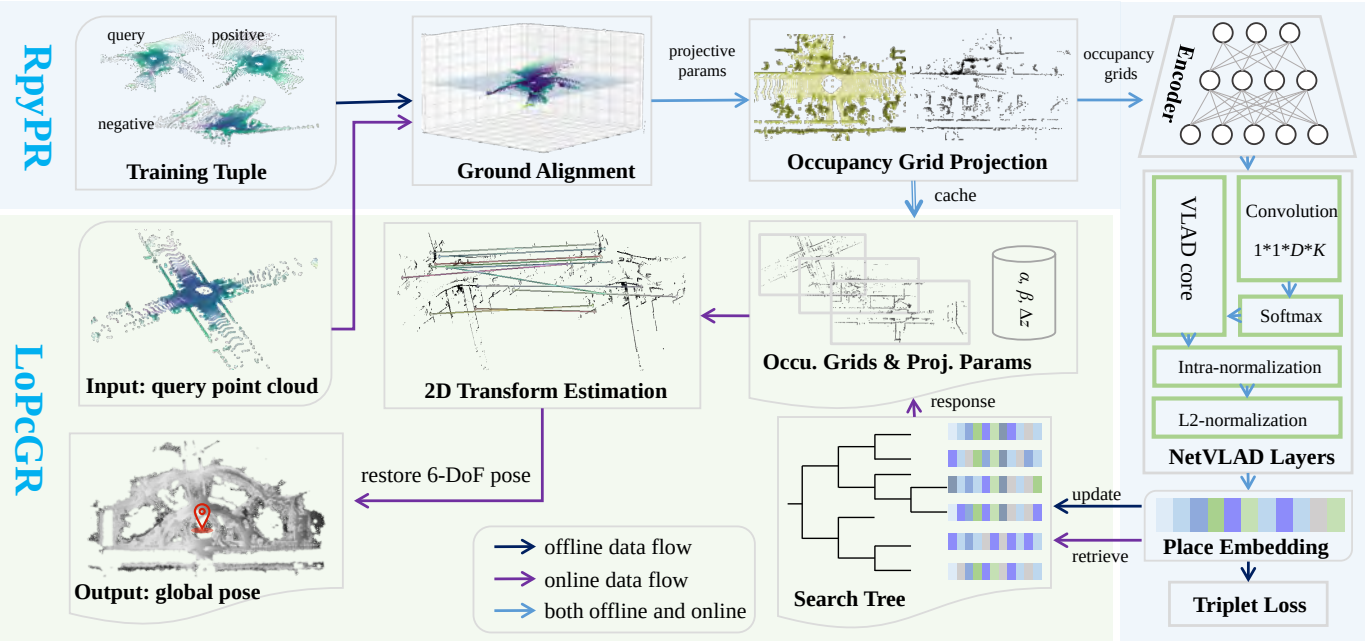


Fig. 2: Overview of our framework. In the offline RpyPR training phase, sampled query-positive-negative tuples are aligned to the ground to derive projection parameters. With these parameters, 2D occupation grids are projected from the 3D occupancy voxels. Such 2D grids are subsequently fed into a deep encoder and NetVLAD layers to aggregate place embeddings, which are used to update the search tree and network under the supervision of the triplet loss. During online global localization (RpyPR recognition and LoPcGR registration), the query obtains its place embedding and employs it to retrieve the response and the corresponding occupancy grids from the gallery. At last, by estimating the transformation between the 2D grids and combining the projection parameters, the full 6-DoF pose can be determined.

traditionally PFH [32] and FPFH [33] are employed to extract features of salient points to establish the corresponding point pairs, while it is still difficult to sustain the good performance in object-level registration to large-scale sparse point clouds. Recent learning-based approaches learn feature representations of salient points in the implicit feature space from raw point clouds end-to-end employing deep neural networks [28], [29], [35], [43]–[48]. Among them, early learning-based ones mostly focused on the local registration problem, such as PointLK [43], Deep Closest Point [44], PRNet [45], and VRNet [46]. In [28], Huang *et al.* were the first to focus on the global registration problem for low-overlap point clouds, arguing that the key to solving this problem is learning from which feature points to sample. They voxelized the point cloud and extracted their features based on a PointNet encoder, and subsequently encoded feature points from the overlap region by overlap-attention modules. By encoding pair-wise distances and triplet-wise angles in Transformer-like blocks, Qin *et al.* proposed GeoTransformer to learn geometric features for robust superpoint matching [35]. Using a similar attention mechanism, Arnold *et al.* [29] fed point-wise features from the base feature encoder into a module with self-cross attention, thus aggregating both foreground and background contexts simultaneously.

III. Methodology

A. Overview

As illustrated in Figure 2, our full framework is composed of two parts: RpyPR and LoPcGR. RpyPR is carried out in two stages: offline training and online recognition, while LoPcGR only works in online registration. During RpyPR's offline training phase, query-positive-negative tuples are first sampled from the priori point cloud map. Then, their attitudes (α, β, and Δz) relative to the ground are estimated by ground segmentation and plane fitting. Afterwards, according to their associated attitudes, the tuples are converted to probabilistic voxels with the ground as a reference and are further projected to occupancy grids. These grids are fed into a deep encoder and NetVLAD layers to construct place embeddings, and are also cached in the gallery together with the associated projection parameters for follow-up retrieval and registration. At last, these place embeddings flow forward to update the search tree, and the network's parameters are updated under the supervision of the triplet loss. During online global localization (RpyPR recognition and LoPcGR registration), we first get the projective parameters, occupancy grid, and place embedding of the query, and then retrieve the response candidate from the search tree. Subsequently, the occupancy grid of the response is matched with that of the query to obtain the 2D transformation parameters (Δx, Δy, γ). Eventually, the global 6-DoF pose of the query in the response is thereby recovered by integrating the 3D-to-2D projection parameters and the 2D-to-2D transformation values.

B. RpyPR: Roll-pitch-yaw Invariant Place Recognition

Instead of directly projecting point clouds to the bird's-eye view, we build probability voxels and project probability grids to establish place embeddings. Such a choice can offer three distinct advantages. Firstly, projecting point clouds directly onto a bird's-eye view often results in significant noise, which

can hinder the establishment of stable feature associations for follow-up transform estimation. Secondly, errors in estimating roll and pitch angles are inevitable. The probability grid representation allows for accommodating attitude estimation errors. Lastly, the method of probability accumulation naturally reflects the richness of ground objects and encodes associated place features, thereby enhancing the learning of place embedding.

1) Occupancy Grid Generation: When the state ($\boldsymbol{s}$) of the sensor is known, modeling the environment according to the sensor measurements $\mathbb{Z}_{1:t} = \{z_1, z_2, \ldots, z_t\}$ (t is the timestamp/index of the last echo) is called a mapping problem [49], [50]. Due to the measuring noises, mapping is usually carried out in a filtering manner. In range data-based mapping (e.g., LiDAR-based, radar-based, or depth camera-based mapping), a typical idea is to divide the space into voxels ($\mathbb{V} = \{v_i\}, i \in Z^+$) to organize the sparse data. The mapping is thereby transformed into a problem of finding the joint probability distribution of the voxels,

$$p(\mathbb{V}|\mathbb{Z}_{1:t}, \boldsymbol{s}). \tag{1}$$

One problem to estimate this probability is that its dimension increases exponentially with the map. To mitigate this, it is usually assumed that all voxels are independent of each other, thus simplifying Eq. 1 to,

$$p(\mathbb{V}|\mathbb{Z}_{1:t}, \boldsymbol{s}) = \prod_i p(v_i|\mathbb{Z}_{1:t}, \boldsymbol{s}). \tag{2}$$

Suppose that for a specific voxel v_i, there is a binary state, namely *occupied* o or *free* $\overline{o}$. When measurements are provided, the occupied probability of the voxel is,

$$p(o|\mathbb{Z}_{1:t}) = \frac{p(z_t|o, \mathbb{Z}_{1:t-1})p(o|\mathbb{Z}_{1:t-1})}{p(z_t|\mathbb{Z}_{1:t-1})}, \tag{3}$$

where we omit the known state in the condition for simplicity. With Markov assumption $p(z_t|o, \mathbb{Z}_{1:t-1}) = p(z_t|o)$, apply Bayes' rule on Eq. 3 yields,

$$p(o|\mathbb{Z}_{1:t}) = \frac{p(o|z_t)p(z_t)}{p(o)} \frac{p(o|\mathbb{Z}_{1:t-1})}{p(z_t|\mathbb{Z}_{1:t-1})}. \tag{4}$$

Likewise, conducting the above procedure again on the probability of $\overline{o}$ produces,

$$p(\overline{o}|\mathbb{Z}_{1:t}) = \frac{p(\overline{o}|z_t)p(z_t)}{p(\overline{o})} \frac{p(\overline{o}|\mathbb{Z}_{1:t-1})}{p(z_t|\mathbb{Z}_{1:t-1})}. \tag{5}$$

Dividing the two ends of Eq. 4 and Eq. 5 results in,

$$\frac{p(o|\mathbb{Z}_{1:t})}{p(\overline{o}|\mathbb{Z}_{1:t})} = \frac{p(o|z_t)}{p(\overline{o}|z_t)} \frac{p(\overline{o})}{p(o)} \frac{p(o|\mathbb{Z}_{1:t-1})}{p(\overline{o}|\mathbb{Z}_{1:t-1})}, \tag{6}$$

Further taking the logarithm of Eq. 6, we have,

$$\begin{aligned} l_t(o) &= \log \frac{p(o|z_t)}{p(\overline{o}|z_t)} + \log \frac{p(\overline{o})}{p(o)} + l_{t-1}(o), \\ l_t(o) &= \log \frac{p(o|\mathbb{Z}_{1:t})}{p(\overline{o}|\mathbb{Z}_{1:t})},\ l_{t-1}(o) = \log \frac{p(o|\mathbb{Z}_{1:t-1})}{p(\overline{o}|\mathbb{Z}_{1:t-1})}. \end{aligned} \tag{7}$$

Note that in Eq. 7, the sum of $p(o|z_t)$ and $p(\overline{o}|z_t)$ is 1, and so are $p(o)$ and $p(\overline{o})$. Moreover, $p(o|z_t)$ is a fixed value related to the sensor's measurement noise only, and $p(o)$ represents the prior of the voxel state. Therefore, at this point, we have obtained a complete recursive update rule for a voxel's state.

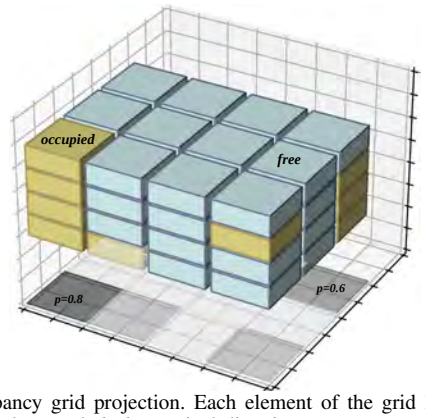


Fig. 3: Occupancy grid projection. Each element of the grid holds the sum probability of the voxels in its vertical direction.

When modeling the environment from a point cloud aligned with the ground, we divide the space into 3D voxels first and update the corresponding voxels point by point according to Eq. 7. After obtaining the 3D voxel representation, feature extraction can be carried out with the help of tools such as 3D convolution. However, considering that on the one hand, it will bring a huge computing load, on the other hand, it is still difficult to learn place embeddings directly from the occupancy voxels. Inspired by the map representation of 2D SLAM [51], we consider generating the occupancy grid corresponding to the point cloud from the bird's-eye view. Specifically, for the voxel set of a point cloud, we generate a 2D grid corresponding to its spatial range on the horizontal plane. Then, for each element of the grid, we take the sum of the probabilities of all voxels accumulated in the vertical direction as its value. In this way, each 2D grid will reflect the richness of ground objects in the corresponding vertical direction, and express the unique characteristics of the place macroscopically. As an intuitive example, our occupancy grid projection is illustrated in Figure 3, where voxels in yellow stand for occupied space while ones in blue imply free there. After projection, a 2D patch in a darker color means that more obstacles are encountered by the LiDAR scanner in its vertical direction.

2) Place Embedding Encoding: Thanks to the compact representation of point clouds with occupancy grids, we can learn global place embeddings conveniently. In specific, those grids are first fed into a commonly used convolution encoder (e.g., VGG [52] etc.) to extract deep semantic features. Then, with the help of the learnable image retrieval technology, NetVLAD [19], we further cluster the deep semantic features with the learning goal of minimizing the intra-class distance and maximizing the inter-class distance. There are two kinds of parameters to be learned by NetVLAD layers. One is the clustering centers of all feature classes ($\{\boldsymbol{c}_k\}, \boldsymbol{c}_k \in \mathbb{R}^D, k \in \{1, 2, \ldots, K\}$, K is the number of class centers). The other is the weights ($\{\boldsymbol{w}_k\}$ and biases $\{\boldsymbol{b}_k\}, \boldsymbol{w}_k, \boldsymbol{b}_k \in \mathbb{R}^D$) that features are given by soft-assignment. For features, $\{\boldsymbol{x}_i\}, \boldsymbol{x}_i \in \mathbb{R}^D$, encoded by the encoder, the associated soft-assignment function which assigns $\boldsymbol{x}_i$ to $\boldsymbol{c}_k$ is defined as,

$$\lambda(\boldsymbol{x}_i) = \frac{e^{\boldsymbol{w}_k^T \boldsymbol{x}_i + \boldsymbol{b}_k}}{\sum_{k'=1}^{K} e^{\boldsymbol{w}_{k'}^T \boldsymbol{x}_i + \boldsymbol{b}_{k'}}}. \tag{8}$$

For the k-th class, its associated feature $\boldsymbol{f}_k$ of the given input $\{\boldsymbol{x_i}\}$ is eventually weight-averaged via,

$$\boldsymbol{f}_k = \sum_i \lambda(\boldsymbol{x}_i)(\boldsymbol{x}_i - \boldsymbol{c}_k). \tag{9}$$

Stacking $\boldsymbol{f}_k, k \in \{1, 2, \ldots, K\}$ leads to a VLAD matrix $\boldsymbol{V}$ with dimensions of $K \times D$. When querying, one always wishes to receive a response with a low delay. Although $\boldsymbol{V}$ can be directly utilized as a distinctive place embedding to construct a search tree (e.g., KD-Tree, etc.), building and querying such a high-dimensional tree is not efficient enough. Therefore, we further append a fully-connected layer after the NetVLAD layers to output compressed place embeddings $\{\boldsymbol{v}\}$ for the search tree construction and query.

3) Metric Learning: During training, for each query sample, we randomly sample positive samples at places close to its measurement location and negative samples far away from it from the priori point cloud map to form triplet tuples as network inputs. After backbone encoding and NetVLAD aggregation, the network parameters are updated by supervised learning of a triplet loss. In specific, for the triple output $(\boldsymbol{v}_q, \boldsymbol{v}_p, \boldsymbol{v}_n)$ where $\boldsymbol{v}_q, \boldsymbol{v}_p$, and $\boldsymbol{v}_n$ are the embeddings of the query, positive, and negative samples respectively, its corresponding loss is,

$$L(\boldsymbol{v}_q, \boldsymbol{v}_p, \boldsymbol{v}_n) = \max(\|\boldsymbol{v}_q - \boldsymbol{v}_p\| - \|\boldsymbol{v}_q - \boldsymbol{v}_n\| + \mu, 0), \tag{10}$$

where $\|\cdot\|$ returns the l-2 norm of the associated vector, and μ refers to the margin which is a hyperparameter to render the network learn features with a larger distance between the query and the negative than that between the query and the positive. Note that the network can be trained end-to-end solely with this loss term.

C. LoPcGR: Low-overlap Point Cloud Global Registration

1) Roll-Pitch Angles and the Vertical Translation: On unmanned ground vehicles such as robots and autonomous cars, the query and response point clouds collected at the same place usually have a common reference plane, that is, the ground. Therefore, we consider estimating and aligning the ground planes from those point clouds, so as to indirectly estimate the roll-pitch angles ($\alpha,\ \beta$) and the vertical translation (Δz).

Specifically, for the point cloud $\{\boldsymbol{p}_i\}$ ($\boldsymbol{p}_i \in \mathbb{R}^3$ indexed by i), which is inclined relative to the ground, we need to screen out the ground points first. To lower the computation load, we only keep a certain range of points around the vehicle for ground estimation. This strategy also improves the accuracy of plane fitting due to the fact that the outliers are growing with distance. It is not difficult to infer that the normal vectors of the ground points have approximately the same or opposite directions. Therefore, we estimate the normal vector of each point and obtain the corresponding normal-to-z angle between the normal vector and the positive z-axis of the LiDAR frame $\mathcal{F}_l$. Subsequently, these points are assigned into 18 bins with sizes of 10° growing from 0° to 180° according to their normal-to-z angles. After that, if the sum of the angles corresponding to the two bins with the highest number of points is close to 180°, it is considered that these two bins contain most of the ground points, denoted by $\{{}^g\boldsymbol{p}_i\}$.

Algorithm 1 Estimation of $\alpha,\ \beta$, and Δz

Input: Point clouds of query ($\mathbb{P}_q$) and response ($\mathbb{P}_r$)
Output: Roll-pitch angles ($\alpha,\ \beta$), and z-translation (Δz)

1: **for** $\mathbb{P} \in \{\mathbb{P}_q, \mathbb{P}_r\}$ **do**
2: Filter $\mathbb{P}$ to retain points which are in a certain range ($20m$ in our setting), resulting in $\mathbb{P}'$ with N points
3: Initialize a count histogram $\mathbb{H}$ with 18 bins growing from 0° to 180°
4: Initialize an index container $\mathbb{I}$ with a size of 18
5: Estimate the normals of $\mathbb{P}'$, yielding $\mathbb{N}'$
6: **for all** $i \in [0, 1, \ldots, N-1]$ **do**
7: $\boldsymbol{n}_i^l = \mathbb{N}'[i]$
8: $\theta = \Phi(\boldsymbol{n}_i^l, \boldsymbol{n}_{z_l^+}^l)$, $\Phi(\cdot)$ returns the angle of vectors in $[0^\circ, 180^\circ)$, $\boldsymbol{n}_{z_l^+}^l$ points to the positive z-axis of $\mathcal{F}_l$
9: $j = \theta$ // 10°, // means floor division
10: $\mathbb{H}[j] \longrightarrow \mathbb{H}[j] + 1$
11: Insert i into $\mathbb{I}[j]$
12: **end for**
13: Find the top-2 bins with greatest counts from $\mathbb{H}$, indexed by m, n
14: Get possible ground points ${}^g\mathbb{P} = \{{}^g\boldsymbol{p}\}$ from $\mathbb{P}$ using indices in $\mathbb{I}[m]$ and $\mathbb{I}[n]$
15: Assume the ground plane is defined as Eq. 11
16: Estimate $\boldsymbol{\rho}^*$ by Eq. 12 under RANSAC
17: Get the normal of the ground in $\mathcal{F}_l$, i.e., $\boldsymbol{n}_g^l = \frac{\boldsymbol{\rho}_{1:3}^*}{\|\boldsymbol{\rho}_{1:3}^*\|}$
18: Get the rotation $\boldsymbol{R}_l^g$ and the translation $\boldsymbol{t}_l^g$ by Eq. 13 and Eq. 16, respectively
19: **end for**
20: Repeat line 1∼19, resulting in $\boldsymbol{R}_{l_q}^g$ ($\boldsymbol{R}_{l_r}^g$) and $\boldsymbol{t}_{l_q}^g$ ($\boldsymbol{t}_{l_r}^g$)
21: Get $\boldsymbol{T}_q^r$ via Eq. 17 with $\boldsymbol{R}_{l_q}^g$, $\boldsymbol{R}_{l_r}^g$, $\boldsymbol{t}_{l_q}^g$, and $\boldsymbol{t}_{l_r}^g$
22: Recover α and β from the left-top 3×3 sub-matrix of $\boldsymbol{T}_q^r$ by rotation matrix to Euler angles conversion
23: $\Delta z = \boldsymbol{T}_q^r(2,3)$, $\boldsymbol{T}_q^r(2,3)$ is the element of $\boldsymbol{T}_q^r$ at the third row and the fourth column
24: **return** $\alpha,\ \beta$, and Δz

When $\{{}^g\boldsymbol{p}_i\}$ is screened out, the ground plane fitting can be cast as a least square estimation problem under the RANSAC [53] strategy. The plane function and the corresponding fitting objective are accordingly defined as,

$$\boldsymbol{\rho} \cdot [{}^g\boldsymbol{p}^T, 1]^T = 0, \tag{11}$$

$$\boldsymbol{\rho}^* = \arg\min_{\boldsymbol{\rho}} \sum_i \frac{1}{2} \left(\frac{\boldsymbol{\rho} \cdot [{}^g\boldsymbol{p}_i^T, 1]^T}{\|\boldsymbol{\rho}_{1:3}\|} \right)^2, \tag{12}$$

where $\boldsymbol{\rho} \in \mathbb{R}^4$ represents the parameter vector of the ground plane, ${}^g\boldsymbol{p}$ is a point lies strictly on that plane, and $\boldsymbol{\rho}_{1:3}$ is the vector with the first three elements of $\boldsymbol{\rho}$.

Conducting the above steps yields the normal vector of the ground ($\boldsymbol{n}_g^l = \frac{\boldsymbol{\rho}_{1:3}^*}{\|\boldsymbol{\rho}_{1:3}^*\|}$) under the LiDAR frame $\mathcal{F}_l$. Denote the normal vector which points to the positive z-axis of $\mathcal{F}_l$ by $\boldsymbol{n}_z^l$. The rotation matrix $\boldsymbol{R}_l^g$ that transforms $\boldsymbol{n}_g^l$ into the ground coordinate system $\mathcal{F}_g$ can be obtained by Rodrigues' rotation formula,

$$\boldsymbol{R}_l^g = \boldsymbol{I} + (\sin\eta)\lfloor \boldsymbol{c} \rfloor_\times + (1 - \cos\eta)\lfloor \boldsymbol{c} \rfloor_\times^2, \tag{13}$$

where $\boldsymbol{I}$ is a 3×3 identity matrix,

$$\boldsymbol{c} = \begin{bmatrix} c_1 & c_2 & c_3 \end{bmatrix}^T = \boldsymbol{n}_g^l \times \boldsymbol{n}_z^l, \tag{14}$$

$$\lfloor \boldsymbol{c} \rfloor_\times = \begin{bmatrix} 0 & -c_3 & c_2 \\ c_3 & 0 & -c_1 \\ -c_2 & c_1 & 0 \end{bmatrix}, \tag{15}$$

and η is the rotation angle from $\boldsymbol{n}_g^l$ to $\boldsymbol{n}_z^l$ along $\boldsymbol{c}$.

Correspondingly, the vertical translation $\boldsymbol{t}_l^g$ from $\mathcal{F}_l$ to $\mathcal{F}_g$ is,

$$\boldsymbol{t}_l^g = \begin{bmatrix} 0 & 0 & \frac{\boldsymbol{\rho}_4}{\|\boldsymbol{\rho}_{1:3}\|} \end{bmatrix}. \tag{16}$$

Consequently, let the compound transformation matrix between the query frame $\mathcal{F}_{l_q}$ and $\mathcal{F}_g$ be $\boldsymbol{T}_{l_q}^g$ $(\boldsymbol{R}_{l_q}^g, \boldsymbol{t}_{l_q}^g)$, and the transformation between the response frame $\mathcal{F}_{l_r}$ and $\mathcal{F}_g$ be $\boldsymbol{T}_{l_r}^g$ $(\boldsymbol{R}_{l_r}^g, \boldsymbol{t}_{l_r}^g)$. $\boldsymbol{T}_q^p$ of $\mathcal{F}_{l_q}$ in $\mathcal{F}_{l_r}$ can thereby be obtained accordingly via,

$$\boldsymbol{T}_q^r = \begin{bmatrix} \boldsymbol{R}_q^r & \boldsymbol{t}_q^r \\ \boldsymbol{0}^T & 1 \end{bmatrix} = (\boldsymbol{T}_{l_r}^g)^{-1} \boldsymbol{T}_{l_q}^g = \begin{bmatrix} \boldsymbol{R}_{l_r}^g & \boldsymbol{t}_{l_r}^g \\ \boldsymbol{0}^T & 1 \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{R}_{l_q}^g & \boldsymbol{t}_{l_q}^g \\ \boldsymbol{0}^T & 1 \end{bmatrix}. \tag{17}$$

Note that $\boldsymbol{T}_q^r$ is just 3-DoF determined so far since we ignored the horizontal translation and rotation in the foregoing discussion. At present, we can infer the Euler angles α and β from the left-top rotation matrix of $\boldsymbol{T}_q^r$ and obtain Δz directly while making $\Delta x, \Delta y$ and γ remain to be determined. It should be also noted that since matrix multiplication does not commute, the order of the axes which one rotates about and the internal/external rotation manner will affect the result. For this reason, we follow the convention of external rotation that first rotates about the x-axis, then the y-axis, and finally the z-axis.

For a better understanding, the estimation of α, β, and Δz is formally defined in Algorithm 1.

2) Horizontal Translations and the Yaw Angle: According to Section III-C1, the query frame $\mathcal{F}_{l_q}$ and response frame $\mathcal{F}_{l_r}$ can be aligned to the shared ground with associated projective parameters. Thus, their corresponding two-dimensional occupancy grids can be generated according to the pipeline described in Section III-B1. Such a representation allows us can estimate the horizontal translations (Δx, Δy) and yaw angle (γ) by resorting to image processing techniques. Ideally, our purpose here is essentially a two-dimensional rigid transformation estimation problem. However, due to the noise in the estimation of the roll and pitch angles, we instead regard the relationship between the points on the query occupancy grid $\{\boldsymbol{u}_j^q\}$ and those on the response one $\{\boldsymbol{u}_k^r\}$ ($\boldsymbol{u}_j^q, \boldsymbol{u}_k^r \in \mathbb{R}^2$ indexed by j, k) approximately conforms to an affine transformation,

$$\boldsymbol{u}_k^r = \pi(\boldsymbol{u}_j^q \mid \Delta x, \Delta y, \gamma) + \begin{bmatrix} \delta_x \\ \delta_y \end{bmatrix} \tag{18}$$

$$= s \cdot \begin{bmatrix} \cos\gamma & -\sin\gamma \\ \sin\gamma & \cos\gamma \end{bmatrix} \boldsymbol{u}_j^q + \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} + \begin{bmatrix} \delta_x \\ \delta_y \end{bmatrix}, \tag{19}$$

where $\pi(\cdot)$ transforms a grid point in the query frame to the response frame, s is the affine scale, δ_x and δ_y are residuals of transformation results.

The key to estimating $\Delta x, \Delta y, \gamma$, and s is to establish the correspondences of point pairs between the two occupancy grids. To achieve this goal, we extract the SURF [54] corners from the occupancy grids, and screen the matching pairs with high similarity according to Lowe's criterion [55]. As a result, the problem of estimating 2D transformation parameters is converted to,

$$\{\Delta x, \Delta y, \gamma, s\}^* = \underset{\Delta x, \Delta y, \gamma, s}{\arg\min} \sum_{j \sim k} \frac{1}{2} \left(\|\boldsymbol{u}_k^r - \pi(\boldsymbol{u}_j^q)\| \right)^2, \tag{20}$$

where $j \sim k$ means a valid point pair with correspondences.

Guided by Eq. 20, we estimate the transformation parameters with RANSAC [53]. When the final estimated scale s is close to 1 (we empirically set the difference threshold as 0.1 via experiments), it is considered that the acceptable two-dimensional transformation parameters are obtained. According to the image resolution, these parameters are transformed back to the physical measurement space, and thus the estimates of $\Delta x, \Delta y, \gamma$ of $\mathcal{F}_{l_q}$ in $\mathcal{F}_{l_r}$ are obtained.

At this point, combined with the roll-pitch angles (α and γ) and the vertical translation (Δz), the full 6-DoF pose $(\boldsymbol{R}, \boldsymbol{t})$ of the query frame $\mathcal{F}_{l_q}$ in the response frame $\mathcal{F}_{l_r}$ can be obtained by the following transformation,

$$\boldsymbol{t} = \begin{bmatrix} \Delta x & \Delta y & \Delta z \end{bmatrix}^T, \tag{21}$$

$$\boldsymbol{R} = \boldsymbol{R}_z(\gamma)\boldsymbol{R}_y(\beta)\boldsymbol{R}_x(\alpha) \tag{22}$$

$$= \begin{bmatrix} c\beta \cdot c\gamma & s\alpha \cdot s\beta \cdot c\gamma - c\alpha \cdot s\gamma & c\alpha \cdot s\beta \cdot c\gamma + s\alpha \cdot s\gamma \\ c\beta \cdot s\gamma & s\alpha \cdot s\beta \cdot s\gamma + c\alpha \cdot c\gamma & c\alpha \cdot s\beta \cdot s\gamma - s\alpha \cdot c\beta \\ -s\beta & s\alpha \cdot c\beta & c\alpha \cdot c\beta \end{bmatrix},$$

where $\boldsymbol{R}_\psi(\cdot)$ returns the basic rotation matrix along the corresponding principle axis ψ, $\psi \in \{x, y, z\}$, $s\alpha$ and $c\alpha$ are the abbreviations of $\sin\alpha$ and $\cos\alpha$ (the same for β and γ).

IV. Experiment

A. Setup

1) Datasets: In this section, we demonstrate the effectiveness of our RpyPR and LoPcGR on two point cloud datasets, KITTI and NCLT, which are widely used for LiDAR-related algorithm evaluations. The KITTI dataset [37], [38] includes gray/color binocular images, 64-beam Velodyne point clouds, and synchronized GPS-IMU readings. Its point clouds, odometry ground truth, and GPS coordinates were employed to train and evaluate our algorithms. The NCLT dataset [39] is collected from a mobile robot equipped with a 32-beam LiDAR. Like KITTI, NCLT also contains real-time kinematic positioning GPS coordinates.

Data preparation for RpyPR. The sequences "00∼02", "04∼07", and "10" of KITTI Odometry were used for training, and "08∼09" were treated as test sequences. Since all NCLT sequences were collected from the same geographical space and the time spans among its sequences are much longer than KITTI, to investigate the stabilities of the algorithms against scenario variations over time, we took NCLT's "2012-01-08" as the training sequence and its "2013-04-05" as the test sequence. Additionally, since sequential point clouds are highly similar, to ensure the diversity of data, we first extracted the raw sequences every 5/10 frames to yield sub-sequences with lower frequencies. After that, these training/test sub-sequences were randomly split into gallery sets and query

sets at a ratio of 4:1 according to the ground truth of GPS positions. In splitting the training sets, the sub-sequences with extracting rates of 5 were used, while in splitting the test sets, both the extracting rates of 5 and 10 were included and the resulting sets were marked as "Easy" and "Hard" accordingly. Consequently, "KITTI-Train, NCLT-Train, KITTI-Test-Easy, KITTI-Test-Hard, NCLT-Test-Easy", and "NCLT-Test-Hard" were yielded for training and test.

Data preparation for LoPcGR. The difficulty of global registration is related to the difference in poses. To evaluate the ability of LoPcGR in more detail, we classified the registration samples of KITTI Odometry's 08∼09 sequences into "GR-Easy", "GR-Medium", and "GR-Hard" according to the distances between two point clouds to be registered of 0∼5m, 5∼10m, and 10∼15m, respectively. In addition, since the point clouds of these test sets were almost gathered from approximately horizontal attitudes, to comprehensively evaluate the ability in 6-DoF estimation, we manually applied a roll-pitch-yaw transformation from the normal distribution with a mean of 10° and a standard deviation of 2° to each frame to be registered in test sets.

2) Metrics: Recall Rate (RR). The recall rate is one of the indicators that can best reflect the performance of a place recognition algorithm. It measures the ability of the algorithm to successfully detect similar candidates from the gallery according to the incoming query. The higher the recall rate, the better the algorithm's recognition performance. Denote the number of correctly identified positive and negative samples in the test set by *TP* and *TN*, while the number of misidentified positive and negative samples by *FP* and *FN* respectively. The recall rate (*RR*) is accordingly defined as,

$$RR = \frac{TP}{TP + FN}. \tag{23}$$

Relative Registration Error. Suppose that the poses of the query frame $\mathcal{F}_q$ and the response frame $\mathcal{F}_r$ relative to the world coordinate system $\mathcal{F}_w$ are $\boldsymbol{T}_q^w$ and $\boldsymbol{T}_r^w$, respectively. The ground truth of the relative pose can be obtained accordingly by, $\boldsymbol{T}_q^r = (\boldsymbol{T}_r^w)^{-1}\boldsymbol{T}_q^w$. Let the relative pose of $\mathcal{F}_q$ in $\mathcal{F}_r$ estimated by an algorithm be $\hat{\boldsymbol{T}}_q^r$. The difference $\Delta\boldsymbol{T}$ between $\boldsymbol{T}_q^r$ and $\hat{\boldsymbol{T}}_q^r$ is therefore obtained by $\Delta\boldsymbol{T} = (\boldsymbol{T}_q^r)^{-1}\hat{\boldsymbol{T}}_q^r$. As a result, the translation error (*TE*) and the rotation error (*RE*) of the registration can be measured by,

$$\Delta\boldsymbol{T} = \begin{bmatrix} \Delta\boldsymbol{R} & \Delta\boldsymbol{t} \\ \boldsymbol{0}^T & 1 \end{bmatrix}, \tag{24}$$

$$TE = \|\Delta\boldsymbol{t}\|, \tag{25}$$

$$RE = \arccos\frac{\mathrm{Tr}(\Delta\boldsymbol{R}) - 1}{2}, \tag{26}$$

where $\Delta\boldsymbol{R} \in \mathbb{R}^{3\times3}$ and $\Delta\boldsymbol{t} \in \mathbb{R}^3$ are the rotation matrix and translation vector of $\Delta\boldsymbol{T}$, and $\mathrm{Tr}(\cdot)$ returns the trace of the involved matrix.

Success Rate (*SR*). The ratio of the total number of successful tests to that of all tests is defined as the success rate. Both in the evaluation of global registration and global localization, we adopted the metrics of success rate, but their specific meaning and criteria were different. In evaluating global registration, the test with a translation error less than 1.5m and a rotation error less than 5° was regarded as a *success*, otherwise, it was a *failure*. In evaluating global localization, a full trial of successful place recognition and global registration was treated as a *success*, otherwise, it was a *failure*.

3) Implementation: During training, for each query frame, we randomly sampled its positive candidates within a 20-meter radius of its geographical location, and obtained negative samples from locations beyond 50 meters away from the query's location. Then, the triplet inputs were formed with these samples and were fed into the network. To optimize the network parameters, we utilized the SGD optimizer with a momentum of 0.9 and an initial learning rate of 0.0001. The learning rate decayed every 5 epochs with a decaying rate of 0.5. The optimization process spanned 30 epochs.

All experiments were carried out on a workstation with processors of "AMD Epyc 7302 16-core processor × 32" and an "NVIDIA GeForce RTX 3090" graphics card.

B. Results of Place Recognition

In this part, we corroborate the effectiveness of our proposed RpyPR from two aspects. One is its high recall rate, and the other is its superior generalization ability. Besides, RpyPR's time cost is also to be discussed.

1) About Recall Rate: We trained our RpyPR and its counterparts, PointNetVLAD [19], LPD-Net [17], and OverlapTransformer [21] on KITTI-Train and NCLT-Train respectively. Subsequently, these models were employed to evaluate their recall rates on KITTI-Test-Easy, KITTI-Test-Hard, NCLT-Test-Easy, and NCLT-Test-Hard. In addition, a representative of handcraft-based methods, Scan-Context [12], was also compared. The detailed results are provided in Table I.

From Table I, it can be seen that although our RpyPR doesn't have the best results on all test sets of NCLT, its recall rates on non-first test sets are merely less than 1% lower than the champions on those sets. In addition, RpyPR achieves 98.80% (96.17%) of top-1 *RR* and 100% (99.04%) of top-20 *RR* on NCLT-Test-Easy (NCLT-Test-Hard), which is qualified to meet the requirements of high recall rate of global localization. On the KITTI dataset, our RpyPR outperforms by a large margin over its counterparts. In specific, on KITTI-Test-Easy, our RpyPR's top-1 *RR* exceeds the runner-up by 7.5 percentage points. In the more difficult test on "KITTI-Test-Hard", its top-1 *RR* surpasses the runner-up largely by 19.5 percentage points. It can also be found that RpyPR's recall rates on NCLT are higher than that on KITTI. We argue that this performance difference is mainly due to the distribution difference between training and test sets. In essence, although the training and test sequences of NCLT came from different trajectories, they were all collected from the same scene, which makes it easier for models trained on NCLT to perform well on similar distributions. Differently, the training and test sequences of KITTI were completely from different trajectories and scenes, which inevitably increases the difficulty.

2) About Generalization Ability: In practical applications, a place recognition model with superlative generalization ability is highly desired, which will avoid laborious data collection and tedious training. In order to evaluate the generalization

TABLE I: Recall rates of our RpyPR and its counterparts on test sets of NCLT and KITTI.

	NCLT-Test-Easy				NCLT-Test-Hard				KITTI-Test-Easy				KITTI-Test-Hard			
	top-1	top-5	top-10	top-20	top-1	top-5	top-10	top-20	top-1	top-5	top-10	top-20	top-1	top-5	top-10	top-20
Scan-Context [12]	0.9414	0.9713	0.9785	0.9856	0.8947	0.9498	0.9689	0.9737	0.6504	0.8717	0.8982	0.9381	0.3805	0.6372	0.7257	0.8142
PointNetVLAD [16]	**0.9928**	0.9976	**1.0000**	**1.0000**	**0.9665**	**0.9904**	**0.9928**	0.9976	0.8673	0.9690	0.9867	0.9912	0.5398	0.8142	0.8761	0.9204
LPD-Net [17]	0.9797	0.9976	0.9976	**1.0000**	0.9641	0.9880	0.9904	**1.0000**	0.6947	0.8761	0.9425	0.9823	0.3186	0.6549	0.7876	0.8496
OverlapTransformer [21]	0.9115	0.9545	0.9737	0.9809	0.8636	0.9569	0.9665	0.9737	0.8363	0.9292	0.9602	0.9867	0.5221	0.7522	0.8407	0.9027
RpyPR (ours)	0.9880	**1.0000**	**1.0000**	**1.0000**	0.9617	0.9809	0.9856	0.9904	**0.9425**	**0.9912**	**0.9956**	**0.9956**	**0.7345**	**0.9469**	**0.9823**	**1.0000**

TABLE II: Generalization ability comparison. The recall rates below of NCLT-Test-Easy and NCLT-Test-Hard are produced by the models trained on KITTI, while those of KITTI-Test-Easy and KITTI-Test-Hard are generated by the models trained on NCLT.

	NCLT-Test-Easy				NCLT-Test-Hard				KITTI-Test-Easy				KITTI-Test-Hard			
	top-1	top-5	top-10	top-20	top-1	top-5	top-10	top-20	top-1	top-5	top-10	top-20	top-1	top-5	top-10	top-20
PointNetVLAD [16]	0.8720	0.9414	0.9629	0.9761	0.7967	0.8804	0.9139	0.9378	0.6726	0.8496	0.9513	0.9823	0.3274	0.6637	0.7434	0.8673
LPD-Net [17]	0.9294	0.9725	0.9821	0.9868	0.8660	0.9187	0.9545	0.9665	0.4558	0.7035	0.8142	0.9115	0.3009	0.5929	0.6726	0.7611
OverlapTransformer [21]	0.9091	0.9629	0.9785	0.9916	0.8612	0.9450	0.9713	0.9833	0.4956	0.7035	0.7699	0.8673	0.3097	0.5044	0.5929	0.7257
RpyPR (ours)	**0.9892**	**0.9988**	**1.0000**	**1.0000**	**0.9569**	**0.9856**	**0.9904**	**0.9952**	**0.8540**	**0.9779**	**0.9912**	**0.9956**	**0.6106**	**0.8673**	**0.9292**	**0.9646**

capabilities of RpyPR and its opponents, we conducted tests on KITTI-Test-Easy and KITTI-Test-Hard using the models trained on NCLT-Train, and tests on NCLT-Test-Easy and NCLT-Test-Hard using the models trained on KITTI-Train. The obtained results are listed in Table II. It can be seen that the generalization ability of our RpyPR is markedly better than other learning-based approaches. In the place recognition evaluation on NCLT, RpyPR trained on KITTI achieves recall rates of 98.92% at top-1 and 100% at top-20. In the more difficult scenes of KITTI, RpyPR trained from NCLT achieves 85.40% top-1 and 99.56% top-20 recall rates. As for its counterparts, although they produce fairish results on NCLT, their recall rates drop largely when tested on the more challenging KITTI. Such a superior generalization ability of our RpyPR should be attributed to the fact that the probability representation from the bird's-eye view effectively narrows the morphological differences between point clouds in different scenes so that the model trained in one scene can be well qualified for place recognition in other different scenes.

3) About Recognition Efficiency: The time consumption of RpyPR in place recognition is about 105 milliseconds (ms), including 28ms for preprocessing, 27ms for embedding extraction, and 50ms for querying. In global localization, such time efficiency is sufficient for most cases. In SLAM relocalization, since the frame rates of most LiDARs today, such as Velodyne, Livox, etc., are usually about 10 frames per second, and the SLAM backends usually only processes keyframes, our RpyPR is qualified to help SLAM systems eliminate their accumulated errors in time.

C. Results of Global Registration

TABLE III: Success rates of global registration approaches.

Method	GR-Easy	GR-Medium	GR-Hard
FPFH [33]	82.64%	73.14%	47.50%
FastReg [29]	74.93%	56.01%	34.37%
PREDATOR [28]	81.54%	63.43%	44.39%
GeoTransformer [35]	80.44%	61.13%	44.73%
LoPcGR (ours)	**83.75%**	**79.15%**	**49.91%**

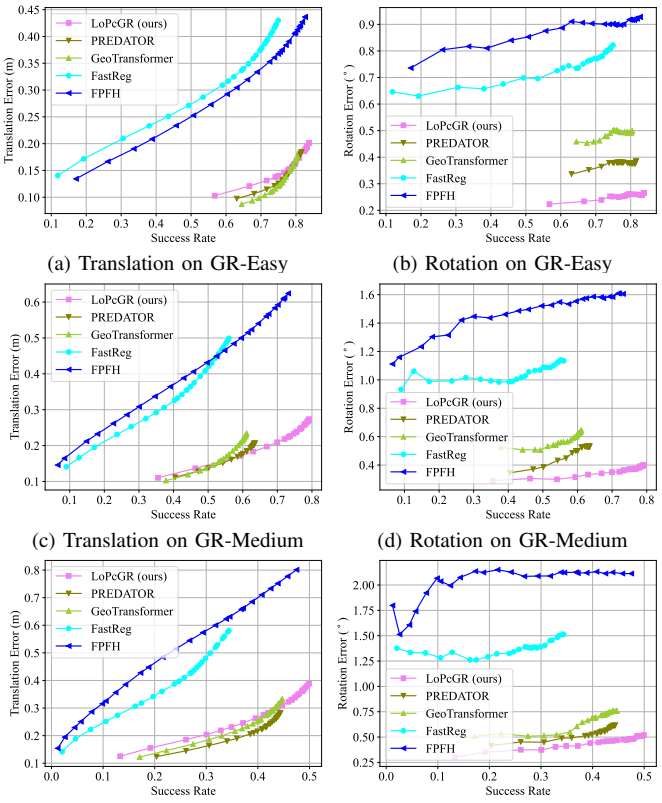
In global registration, we investigate the success rates (*SR*), registration accuracy (*TE*, *RE*), and time costs of LoPcGR and those of its competitors FPFH [33], FastReg [29], PREDATOR [28], and GeoTransformer [35]. The related results are given in Table III, Figure 4, and Table IV.

1) About Success Rate: As observed in Table III, LoPcGR achieves the highest success rates on datasets at different difficulty levels. Especially in the registration tests on "GR-Medium" and "GR-Hard", LoPcGR surpasses the runner-ups by large margins. It should be emphasized that such high success rates on challenging samples are of great significance for global localization and re-localization in SLAM since the query and response point clouds may only share a small overlap. Therefore, a high success rate in global registration becomes the premise for successful global localization, which also ensures that the accumulated errors in SLAM can be eliminated in time.

2) About Registration Accuracy: The accuracy of a registration algorithm can be reflected by its corresponding translation and rotation errors under a specific success rate. However, under the same threshold, the success rates of the algorithms are often different. Therefore, in order to comprehensively evaluate the registration accuracy, we recorded the success rates and the corresponding translation-rotation errors of each algorithm as the threshold varied from fine to coarse. The related results are reported in Figure 4. It can be seen that in terms of the translation accuracy, LoPcGR's is slightly lower than PREDATOR's [28] and GeoTransformer's [35] in the low success rate intervals, but its accuracy is better than theirs in the high success rate intervals. As for the rotation accuracy, our LoPcGR consistently achieves the best results at any success rate and difficulty level. Overall, when achieving the highest success rates of 83.75%, 79.15%, and 49.91% on GR-Easy, GR-Medium, and GR-Hard respectively, LoPcGR demonstrates impressive accuracy, with values of (0.20m, 0.26°), (0.27m, 0.40°), and (0.39m, 0.52°), respectively.

TABLE IV: Time costs of global registration methods. N/A means unavailable.

Method	GPU time cost	CPU time cost
FPFH [33]	N/A	24.53s
FastReg [29]	1.41s	N/A
PREDATOR [28]	1.76s	29.57s
GeoTransformer [35]	0.78s	5.03s
LoPcGR (ours)	N/A	**0.11s**



(a) Translation on GR-Easy (b) Rotation on GR-Easy

(c) Translation on GR-Medium (d) Rotation on GR-Medium

(e) Translation on GR-Hard (f) Rotation on GR-Hard

Fig. 4: Translation and rotation errors of different global registration methods.

3) About Registration Efficiency: When examining the time efficiency of global registration, we evaluated the average time consumed by each method to complete a full trial on the GPU or CPU. The recorded results can be found in Table IV, which includes FPFH and LoPcGR without GPU acceleration, and FastReg for which the CPU time data was unavailable. From Table IV, it can be seen that our LoPcGR has a notable efficiency advantage over the competing approaches. Compared to traditional handcraft feature-based FPFH, LoPcGR achieves a registration acceleration of approximately 220 times. Compared to deep learning-based methods (FastReg, PREDATOR, and GeoTransformer), even compared with their time costs with GPU acceleration, LoPcGR's efficiency remains comparable. Under the premise that all the algorithms to be evaluated run on the CPU, the running efficiency of LoPcGR is about 50 times that of the runner-up.

4) About Qualitative Results: Due to the lack of credible ground truth of inter-frame registration in NCLT, it is infeasible to quantitatively evaluate the registration results on NCLT. To demonstrate the adaptability of LoPcGR for different scenarios, we also provide qualitative results of LoPcGR registration between various difficult samples on NCLT. Under the obtained registration parameters, we transform the query point cloud to the coordinate system of the response point cloud. As shown in Figure 5, LoPcGR can successfully achieve global registration in various difficult cases. The point clouds after registration show good matching at common prominent features (such as street lamp poles, wall corners, flower beds, etc.), demonstrating the correctness of the registration.

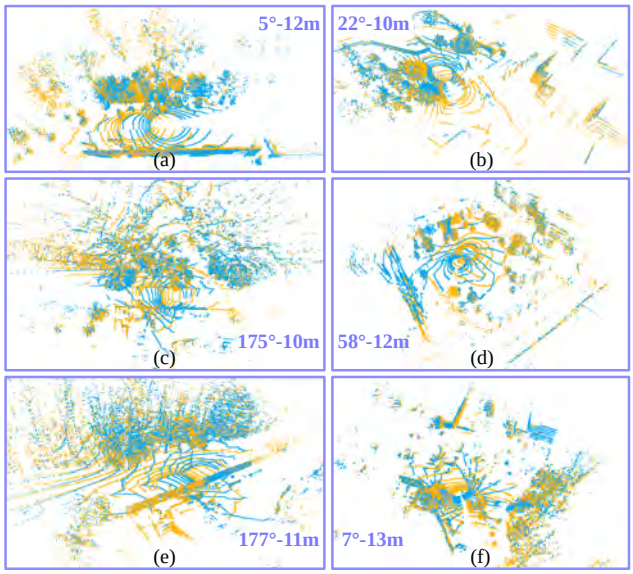
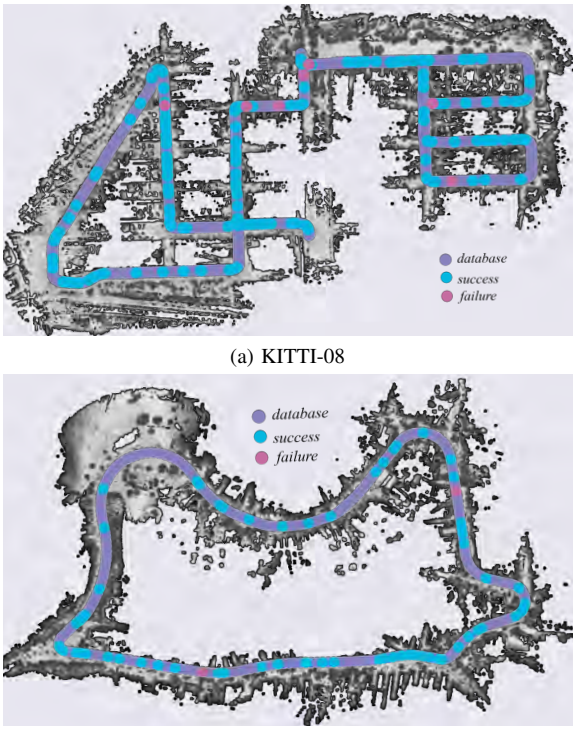


Fig. 5: Typical registered point clouds by LoPcGR on low-overlap cases of NCLT. The query and response point clouds in (a)∼(f) are with large differences in translation, while the ones in (b)∼(e) are with great differences in rotation. The blue and yellow dots indicate the points of the query and response frames, respectively. The annotation "5°-12m" in (a) means that the query and response point clouds are with a 5° difference in rotation and with a 12m difference in translation, so do (b)∼(f).



(a) KITTI-08

(b) KITTI-09

Fig. 6: Global localization on KITTI-08 and KITTI-09.

D. Performance in Real-world Applications

In this subsection, we investigate the practicability of RpyPR and LoPcGR in the practical applications of global localization and SLAM.

1) Performance in Global Localization: We first test the global localization on sequences KITTI-08 and KITTI-09, which were gathered along urban and mountain roads. The

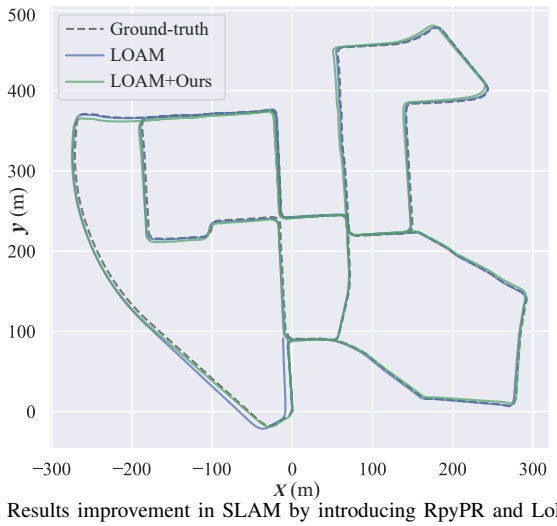


Fig. 7: Results improvement in SLAM by introducing RpyPR and LoPcGR. When driving back to the start, the trajectory estimated by the variant of LOAM that is enhanced by our RpyPR and LoPcGR shows a much better geometric consistency with the ground truth.

model trained on NCLT-Train was utilized to extract features from the gallery to build search trees. During the localization, top 20 candidates were reserved for global registration one by one for geometric verification and the pose of the most similar one which was successfully registered was regarded as the result. As shown in Figure 6, the purple points represent the collection locations of the historical point clouds in the gallery, and the blue and pink points stand for the locations of data collection that are successfully localized and failed, respectively. It can be seen that the proposed RpyPR and LoPcGR can produce pleasing global localization results with strong geometry consistency both along complex urban roads and in challenging mountain environments. According to statistics, our global localization manages to achieve an average success rate of 90.26% in these challenging scenarios.

2) Performance in SLAM: We show another important application of SLAM on KITTI-01. Zhang and Singh's famous LOAM [56], a LiDAR SLAM approach that consists of a front-end odometry and a backend mapping module was taken as the baseline. At LOAM's back-end, we deployed our RpyPR model trained on NCLT to detect the loop closures of the point clouds and resorted to LoPcGR to establish the relative pose constraints between the incoming frame and the historical frame, so as to build and optimize a global pose graph to eliminate the cumulative error. In Figure 7, we present the estimated trajectories, where the gray dotted line, the blue solid line, and the green solid line represent the ground truth, LOAM's trajectory, and the trajectory with our RpyPR and LoPcGR, respectively. It can be seen that the estimated trajectory has good global consistency with the ground truth by introducing RpyPR and LoPcGR, while that estimated by LOAM obviously drifts when driving back to the start point.

E. Ablation Study

To further demonstrate the indispensability of the submodules in LoPcGR, we conducted ablation studies on its α-β-Δz estimation (denoted by LoPcGR-$\alpha\beta\Delta z$) and Δx-Δy-γ estimation (denoted by LoPcGR-$\Delta x\Delta y\gamma$) on the "GR-Easy" test set. Since only three degrees of freedom are determined by LoPcGR-$\alpha\beta\Delta z$ or LoPcGR-$\Delta x\Delta y\gamma$ separately, we took their respective results and LoPcGR's output as initial values, supplemented by NDT registration [24], to refine the global registration and recorded their success rates in Table V.

TABLE V: Ablation study on LoPcGR. LoPcGR-$\alpha\beta\Delta z$ (LoPcGR-$\Delta x\Delta y\gamma$) denotes α-β-Δz estimation (Δx-Δy-γ estimation).

Method	LoPcGR-$\alpha\beta\Delta z$	LoPcGR-$\Delta x\Delta y\gamma$	LoPcGR
Success rate	5%	67%	**84%**

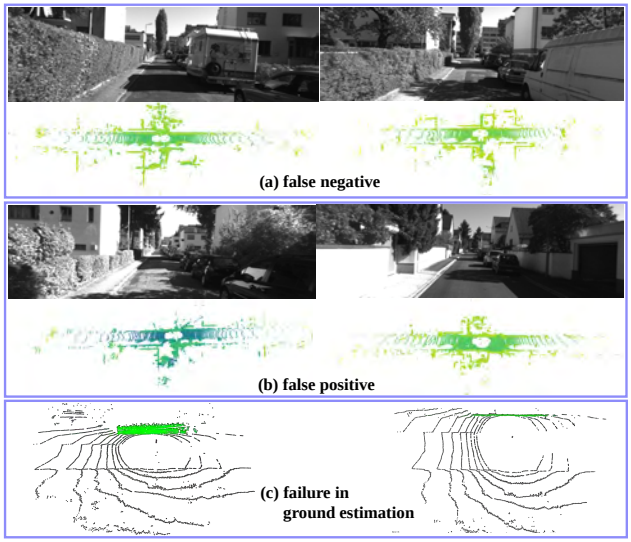


Fig. 8: Typical failure cases. (a) A positive sample is misidentified as negative. (b) A negative sample is misidentified as positive. (c) The points belonging to a fence are fitted as the ground.

As shown in Table V, it can be seen that the success rate is very low when only estimating the initial values of α, β, and Δz. By contrast, successfully estimating Δx, Δy, and γ can significantly improve the success rate. The reason for this phenomenon is that the horizontal translation differences between the point clouds to be registered are generally greater than their vertical translation difference, and their attitude difference in yaw is greater than that in roll and pitch. When LoPcGR-$\alpha\beta\Delta z$ and LoPcGR-$\Delta x\Delta y\gamma$ take effect simultaneously, it can be found that the complete LoPcGR achieves a predominant success rate of global registration. Such a remarkable result verifies the indispensability of LoPcGR's submodules.

F. Failure Case Study

Although the excellent performance of our RpyPR and LoPcGR has been corroborated in the previous experiments, there still exist some extreme cases that are hard to handle.

1) Failure in Place Recognition: In place recognition, the two typical types of failure cases are false negatives and false positives. A false negative results in the omission of a candidate location that should have been detected. As illustrated in Figure 8 (a), two frames of point clouds belonging to the same place are obviously different due to the occlusion of large obstacles nearby. A false positive mainly comes from highly similar places. For example, the two frames of point clouds in Figure 8 (b) are collected in a narrow road, which leads to

a high degree of similarity between them from the bird's-eye view, thus leading to false recognition.

Note that in a global localization system, the tolerance for false negatives is lower than that for false positives. This is because the false positives can be filtered by further geometric verification, while the false negatives will directly lead to the failure of localization. Therefore, it is suggested that more candidates can be appropriately retained to ensure fewer missed detections.

2) Failure in Global Registration: In global registration, we find that one kind of the most typical failures comes from ground estimation. As shown in Figure 8 (c), in some extreme locations, a large number of points whose normal vectors are perpendicular to the ground appear on a fence, which leads the algorithm to misjudge it as the ground. If the computing resources allow, this negative effect is expected to be alleviated by successively fitting the two bins containing the most points as candidates and further conducting geometric verification.

V. Conclusion

To achieve global localization for ground unmanned vehicles in large-scale scenes using multi-beam LiDAR, we study two sub-problems, place recognition and global registration, in this article. For place recognition, we propose to learn the place embeddings with discrimination from the perspective of bird's-eye view, so as to construct a compact embedding gallery and fulfill fast similar candidates retrieval, yielding RpyPR with roll-pitch-yaw invariances. For the global registration of point clouds with low overlaps, we propose LoPcGR which estimates the relative roll-pitch-Δz parameters by fitting and aligning the ground plane from the point clouds and conducts the 2D transformation estimation to determine Δx-Δy-yaw values by matching the key points of the projected occupancy grids, so as to restore the full 6-DoF global pose. We evaluate the proposed RpyPR and LoPcGR comprehensively through a large number of experiments. In terms of recall, success rate, generalization capacity, and accuracy, our approaches achieve advanced results. Especially in the recognition and registration of hard samples, our RpyPR and LoPcGR are far superior to their counterparts in performance. In addition, we also corroborate RpyPR's and LoPcGR's practicability via practical applications of global localization and SLAM in complex scenarios.

References

[1] Y. Li and J. Ibanez-Guzman, "LiDAR for autonomous driving: The principles, challenges, and trends for automotive LiDAR and perception systems," *IEEE Signal Process. Mag.*, vol. 37, no. 4, pp. 50–61, 2020.

[2] H. Yin, L. Tang, X. Ding, Y. Wang, and R. Xiong, "LocNet: Global localization in 3D point clouds for mobile vehicles," in *Proc. IEEE Intell. Vehicles Symp.*, Changshu, China, Jun. 2018, pp. 728–733.

[3] S. Lowry, N. Sünderhauf, P. Newman, J. J. Leonard, D. Cox, P. Corke, and M. J. Milford, "Visual place recognition: A survey," *IEEE Trans. Robot.*, vol. 32, no. 1, pp. 1–19, 2016.

[4] K. A. Tsintotas, L. Bampis, and A. Gasteratos, "The revisiting problem in simultaneous localization and mapping: A survey on visual loop closure detection," *IEEE Trans. Intell. Transport. Syst.*, vol. 23, no. 11, pp. 19 929–19 953, 2022.

[5] Y. Wang, Y. Qiu, P. Cheng, and J. Zhang, "Hybrid CNN-Transformer features for visual place recognition," *IEEE Trans. Circuits Syst. Video Tech.*, vol. 33, no. 3, pp. 1109–1122, 2023.

[6] R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu, "Fast 3D recognition and pose using the viewpoint feature histogram," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Taiwan, China, Oct. 2010, pp. 2155–2162.

[7] A. Aldoma, M. Vincze, N. Blodow, D. Gossow, S. Gedikli, R. B. Rusu, and G. Bradski, "CAD-model recognition and 6DOF pose estimation using 3D cues," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshop*, Barcelona, Spain, Nov. 2011, pp. 585–592.

[8] N. Muhammad and S. Lacroix, "Loop closure detection using small-sized signatures from 3D LiDAR data," in *Proc. IEEE Int. Symp. Safety, Security, Rescue Robot.*, Kyoto, Japan, Nov. 2011, pp. 333–338.

[9] Y. Cui, X. Chen, Y. Zhang, J. Dong, Q. Wu, and F. Zhu, "BoW3D: Bag of words for real-time loop closing in 3D LiDAR SLAM," *IEEE Robot. Autom. Letters*, Early Access, 2022.

[10] L. He, X. Wang, and H. Zhang, "M2DP: A novel 3D point cloud descriptor and its application in loop closure detection," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Daejeon, Korea, Oct. 2016, pp. 231–237.

[11] F. Tombari, S. Salti, and L. Di Stefano, "A combined texture-shape descriptor for enhanced 3D feature matching," in *Proc. IEEE Int. Conf. Image Process.*, Brussels, Belgium, Sept. 2011, pp. 809–812.

[12] 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.*, Madrid, Spain, Oct. 2018, pp. 4802–4809.

[13] X. Xu, H. Yin, Z. Chen, Y. Li, Y. Wang, and R. Xiong, "DiSCO: Differentiable scan context with orientation," *IEEE Robot. Autom. Letters*, vol. 6, no. 2, pp. 2791–2798, 2021.

[14] Y. Wang, Z. Sun, C. Xu, S. E. Sarma, J. Yang, and H. Kong, "LiDAR Iris for loop-closure detection," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Las Vegas, USA, Oct. 2020, pp. 5769–5775.

[15] L. Luo, S.-Y. Cao, B. Han, H.-L. Shen, and J. Li, "BVMatch: Lidar-based place recognition using bird's-eye view images," *IEEE Robot. Autom. Letters*, vol. 6, no. 3, pp. 6076–6083, 2021.

[16] M. A. Uy and G. H. Lee, "PointnetVLAD: Deep point cloud based retrieval for large-scale place recognition," in *Proc. IEEE Conf. Comput. Vis. Patt. Recog.*, Salt Lake City, USA, Jun. 2018, pp. 4470–4479.

[17] Z. Liu, S. Zhou, C. Suo, P. Yin, W. Chen, H. Wang, H. Li, and Y. H. Liu, "LPD-Net: 3D point cloud learning for large-scale place recognition and environment analysis," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Seoul, Korea, Oct. 2019, pp. 2831–2840.

[18] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Patt. Recog.*, Honolulu, Hawaii, Jul. 2017, pp. 652–660.

[19] R. Arandjelović, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD: CNN architecture for weakly supervised place recognition," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 40, no. 6, pp. 1437–1451, 2018.

[20] 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," *Auto. Robots*, vol. 46, no. 1, pp. 61–81, Jan. 2022.

[21] J. Ma, J. Zhang, J. Xu, R. Ai, W. Gu, and X. Chen, "OverlapTransformer: An efficient and yaw-angle-invariant transformer network for LiDAR-based place recognition," *IEEE Robot. Autom. Letters*, vol. 7, no. 3, pp. 6958–6965, 2022.

[22] J. Ma, X. Chen, J. Xu, and G. Xiong, "SeqOT: A spatial-temporal transformer network for place recognition using sequential LiDAR data," *IEEE Trans. Indust. Elect.*, Early Access, 2022.

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

[24] 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.*, Las Vegas, USA, Oct. 2003, pp. 2743–2748.

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

[26] J. Serafin and G. Grisetti, "NICP: Dense normal based point cloud registration," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Hamburg, Germany, Sept. 2015, pp. 742–749.

[27] C. Ulaş and H. Temeltaş, "3D multi-layered normal distribution transform for fast and long range scan matching," *J. Intell. Robot. Syst.*, vol. 71, pp. 85–108, 2013.

[28] S. Huang, Z. Gojcic, M. Usvyatsov, A. Wieser, and K. Schindler, "PREDATOR: Registration of 3D point clouds with low overlap," in *Proc. IEEE Conf. Comput. Vis. Patt. Recog.*, Nashville, TN, USA, Jun. 2021, pp. 4265–4274.

[29] E. Arnold, S. Mozaffari, and M. Dianati, "Fast and robust registration of partially overlapping point clouds," *IEEE Robot. Autom. Letters*, vol. 7, no. 2, pp. 1502–1509, 2022.

[30] K. Fukunaga and P. M. Narendra, "A branch and bound algorithm for computing k-nearest neighbors," *IEEE Trans. Comput.*, vol. 100, no. 7, pp. 750–753, Jul. 1975.
[31] J. Yang, H. Li, and Y. Jia, "Go-ICP: Solving 3D registration efficiently and globally optimally," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Sydney, Australia, Dec. 2013, pp. 1457–1464.
[32] R. B. Rusu, N. Blodow, Z. C. Marton, and M. Beetz, "Aligning point cloud views using persistent feature histograms," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Nice, France, Sept. 2008, pp. 3384–3391.
[33] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (FPFH) for 3D registration," in *Proc. IEEE Int. Conf. Robot. Autom.*, Kobe, Japan, May 2009, pp. 3212–3217.
[34] J. Du, R. Wang, and D. Cremers, "DH3D: Deep hierarchical 3D descriptors for robust large-scale 6DoF relocalization," in *Proc. Euro. Conf. Comput. Vis.*, Glasgow, United Kingdom, Aug. 2020, pp. 744–762.
[35] Z. Qin, H. Yu, C. Wang, Y. Guo, Y. Peng, and K. Xu, "Geometric Transformer for fast and robust point cloud registration," in *Proc. IEEE Conf. Comput. Vis. Patt. Recog.*, New Orleans, LA, USA, 2022, pp. 11 133–11 142.
[36] Y. Wu, Y. Zhang, X. Fan, M. Gong, Q. Miao, and W. Ma, "INENet: Inliers estimation network with similarity learning for partial overlapping registration," *IEEE Trans. Circuits Syst. Video Tech.*, vol. 33, no. 3, pp. 1413–1426, 2023.
[37] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the KITTI vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Patt. Recog.*, Rhode Island, USA, Jun. 2012, pp. 3354–3361.
[38] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Research*, vol. 32, no. 11, pp. 1231–1237, 2013.
[39] N. Carlevaris-Bianco, A. K. Ushani, and R. M. Eustice, "University of Michigan North Campus long-term vision and LiDAR dataset," *Int. J. Robot. Research*, vol. 35, no. 9, pp. 1023–1035, 2015.
[40] J. Wang, M. Zhu, B. Wang, D. Sun, H. Wei, C. Liu, and H. Nie, "KDA3D: Key-point densification and multi-attention guidance for 3D object detection," *Remote Sensing*, vol. 12, no. 11:1895, 2020.
[41] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Neural Info. Proc. Syst.*, Long Beach, CA, USA, Dec. 2017, pp. 6000–6010.
[42] C. Olsson, F. Kahl, and M. Oskarsson, "Branch-and-bound methods for euclidean registration problems," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 31, no. 5, pp. 783–794, 2009.
[43] Y. Aoki, H. Goforth, R. A. Srivatsan, and S. Lucey, "PointNetLK: Robust & efficient point cloud registration using PointNet," in *Proc. IEEE Conf. Comput. Vis. Patt. Recog.*, Long Beach, CA, USA, Jun. 2019, pp. 7156–7165.
[44] Y. Wang and J. Solomon, "Deep closest point: Learning representations for point cloud registration," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Seoul, Korea, Oct. 2019, pp. 3522–3531.
[45] Y. Wang and J. M. Solomon, "PRNet: Self-supervised learning for partial-to-partial registration," in *Proc. Neural Info. Proc. Syst.*, Vancouver, Canada, Dec. 2019, pp. 1–13.
[46] Z. Zhang, J. Sun, Y. Dai, B. Fan, and M. He, "VRNet: Learning the rectified virtual corresponding points for 3D point cloud registration," *IEEE Trans. Circuits Syst. Video Tech.*, vol. 32, no. 8, pp. 4997–5010, 2022.
[47] S. Ren, Y. Zeng, J. Hou, and X. Chen, "CorrI2P: Deep image-to-point cloud registration via dense correspondence," *IEEE Trans. Circuits Syst. Video Tech.*, vol. 33, no. 3, pp. 1198–1208, 2023.
[48] A. Zhu, Y. Xiao, C. Liu, and Z. Cao, "Robust LiDAR-camera alignment with modality adapted local-to-global representation," *IEEE Trans. Circuits Syst. Video Tech.*, vol. 33, no. 1, pp. 59–73, 2023.
[49] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*, 2nd ed. Cambridge, Massachusetts, USA: The MIT Press, 2005.
[50] D. Bagnell, "Occupancy Maps," University of Cambridge, School of Computer Science, Tech. Rep. 16-831-F10, Sept. 2016.
[51] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2D LiDAR SLAM," in *Proc. IEEE Int. Conf. Robot. Autom.*, Stockholm, Sweden, May 2016, pp. 1271–1278.
[52] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
[53] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Communi. ACM*, vol. 24, no. 6, pp. 381–395, Jan. 1981.
[54] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded up robust features," in *Proc. Euro. Conf. Comput. Vis.*, Graz, Austria, May 2006, pp. 404–417.
[55] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, Jan. 2004.
[56] J. Zhang and S. Singh, "LOAM: Lidar odometry and mapping in real-time," in *Proc. Robot. Sci. Syst. Conf.*, California, USA, Jul. 2014, pp. 1–9.

Zhong Wang received the B.S. and M.S. degrees from the School of Surveying and Geo-Informatics, Tongji University, Shanghai, China, in 2016 and 2019, respectively. Starting from 2020, he is pursuing his Ph.D. degree at the School of Software Engineering, Tongji University, Shanghai, China. His research interests include motion planning of mobile robot, SLAM and computer vision.

Lin Zhang (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees from the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China, in 2003 and 2006, respectively. He received the Ph.D. degree from the Department of Computing, The Hong Kong Polytechnic University, Hong Kong, in 2011. From March 2011 to August 2011, he was a Research Associate with the Department of Computing, The Hong Kong Polytechnic University. In Aug. 2011, he joined the School of Software Engineering, Tongji University, Shanghai, China, where he is currently a Full Professor. His current research interests include environment perception of intelligent vehicle, pattern recognition, computer vision, and perceptual image/video quality assessment. He serves as an Associate Editor for IEEE Robotics and Automation Letters, and Journal of Visual Communication and Image Representation. He was awarded as a Young Scholar of Changjiang Scholars Program, Ministry of Education, China.

Shengjie Zhao (Senior Member, IEEE) received the B.S. degree in electrical engineering from the University of Science and Technology of China, Hefei, China, in 1988, the M.S. degree in electrical and computer engineering from the China Aerospace Institute, Beijing, China, in 1991, and the Ph.D. degree in electrical and computer engineering from Texas A&M University, College Station, TX, USA, in 2004. He is currently the Dean of the School of Software Engineering and a Professor with the School of Software Engineering and the School of Electronics and Information Engineering, Tongji University, Shanghai, China. In previous postings, he conducted research at Lucent Technologies, Whippany, NJ, USA, and the China Aerospace Science and Industry Corporation, Beijing. He is a fellow of the Thousand Talents Program of China and an Academician of the International Eurasian Academy of Sciences. His research interests include artificial intelligence, big data, wireless communications, image processing, and signal processing.

Yicong Zhou (Senior Member, IEEE) received the B.S. degree in electrical engineering from Hunan University, Changsha, China, and the M.S. and Ph.D. degrees in electrical engineering from Tufts University, Medford, MA, USA. He is currently a Full Professor and the Director of the Vision and Image Processing Laboratory, Department of Computer and Information Science, University of Macau, Macau, China. His research interests include chaotic systems, multimedia security, computer vision, and machine learning. He serves as an Associate Editor for Neurocomputing, Journal of Visual Communication and Image Representation, and Signal Processing: Image Communication. He is a Co-Chair of the Technical Committee on Cognitive Computing in the IEEE Systems, Man, and Cybernetics Society. He is a Senior Member of the International Society for Optical Engineering (SPIE).