Triplet-Graph: Global Metric Localization Based on Semantic Triplet Graph for Autonomous Vehicles

Weixin Ma, *Graduate Student Member, IEEE*, Shoudong Huang, *Senior Member, IEEE*, and Yuxiang Sun, *Member, IEEE*

***Abstract*—Global metric localization is one of the fundamental capabilities for autonomous vehicles. Most existing methods rely on global navigation satellite systems (GNSS). Some methods relieve the need of GNSS by using 3-D LiDARs. They first achieve place recognition with a pre-built geo-referenced point-cloud database for coarse global localization, and then achieve 3-DoF/6-DoF pose estimation for fine-grained metric localization. However, these methods require accessing point-cloud features and raw point clouds, making them inefficient and hard to be deployed in large-scale environments. To provide a solution to this issue, we propose a global metric localization method with triplet-based histogram descriptors. Specifically, we first convert the input LiDAR point clouds into a semantic graph and describe the vertices in the graph with the proposed descriptor for vertex matching and pose estimation. These vertex descriptors are then selected and aggregated into a global descriptor to decide whether two places correspond to the same place according to a similarity score. Experimental results on the KITTI dataset demonstrate that our method generally outperforms the sate-of-the-art methods.**

***Index Terms*—Autonomous vehicles, global metric localization, place recognition, pose estimation, semantic triplet graph.**

I. Introduction

GLOBAL metric localization refers to finding vehicle positions at the metric level relative to a given map, usually a geo-referenced map, such as Google Map. It is a fundamental capability for autonomous vehicles to navigate in real-world environments. Vision-based global localization has attracted great attentions in recent decades due to the affordability of visual cameras and great advancement of computer vision technologies. However, vision-based methods suffer from the intrinsic limitations of visual cameras, such as changes of illuminations, viewpoints, weathers, and seasons, etc.

Recently, many global localization methods use 3-D LiDARs due to their robustness to the changes of visual appearances and illuminations. The pipeline of the LiDAR-based methods can be generally divided into two steps: 1) Perform place recognition by matching the global descriptors extracted from both the current and off-line LiDAR point clouds; 2) Estimate relative metric poses by registering the current point cloud to the retrieved point cloud from the database. However, these methods require accessing point-cloud features and raw point clouds for place recognition and pose estimation, making them inefficient and hard to be deployed in large-scale environments. To address this issue, some researchers [1], [2] propose to design descriptors to achieve both place recognition and pose estimation. In this way, only point-cloud features are needed to be accessed, so the memory cost is reduced and the efficiency is increased.

However, many existing descriptor-based methods can only achieve 1-DoF/3-DoF localization and might be degraded when significant viewpoint changes happen between the on-line and off-line point clouds. So, in this letter, we propose a novel method called Triplet-Graph to address the above issues. Specifically, we convert a point cloud into a semantic object graph, and design an innovative semantic triplet-based histogram descriptor to embed semantic, topological, and geometric information for each vertex (i.e., an object) in the graph. We extract vertex descriptors from two different point clouds, then perform vertex matching and estimate the relative pose based on the matched vertices. The estimated pose and the descriptors of the vertices are further employed to build global descriptors by simple statistical calculations. The global descriptors are used to decide whether two places correspond to the same place according to their similarity score. The framework of our proposed method is shown in Fig. 1. Our contributions are summarized as follows:

1) We propose novel semantic triplet-based histogram descriptors for global metric localization using 3-D LiDAR point clouds.
2) We design a simple yet effective adding strategy to fast aggregate vertex descriptors into a global descriptor to reduce memory consumption.
3) We demonstrate that our method generally outperforms state-of-the-art methods in terms of both place recognition and pose estimation. Our code is open-sourced.[1]

II. Related Work

The existing global metric localization methods using key-frame-based database can be generally divided into two categories: one-shot and filtering. Our method belongs to one-shot, which means that for the on-line data we only need the point cloud from the current moment. Different from one-shot,

Manuscript received 19 August 2023; accepted 6 January 2024. Date of publication 25 January 2024; date of current version 22 February 2024. This letter was recommended for publication by Associate Editor S. Thakar and Editor A. Banerjee upon evaluation of the reviewers' comments. This work was supported in part by Hong Kong Research Grants Council under Grant 15222523, and in part by the National Natural Science Foundation of China under Grant 62003286. *(Corresponding author: Yuxiang Sun.)*

Weixin Ma is with the Department of Mechanical Engineering, The Hong Kong Polytechnic University, Kowloon, Hong Kong (e-mail: weixin.ma@connect.polyu.hk).

Shoudong Huang is with the Robotics Institute, Faculty of Engineering and Information Technology, University of Technology Sydney, Ultimo, Sydney, NSW 2007, Australia (e-mail: shoudong.huang@uts.edu.au).

Yuxiang Sun is with the Department of Mechanical Engineering, City University of Hong Kong, Kowloon, Hong Kong (e-mail: yx.sun@cityu.edu.hk).

Digital Object Identifier 10.1109/LRA.2024.3358752

[1][Online]. Available: https://github.com/lab-sun/Triplet-Graph

2377-3766 © 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: CITY UNIV OF HONG KONG. Downloaded on February 28,2024 at 07:01:23 UTC from IEEE Xplore. Restrictions apply.

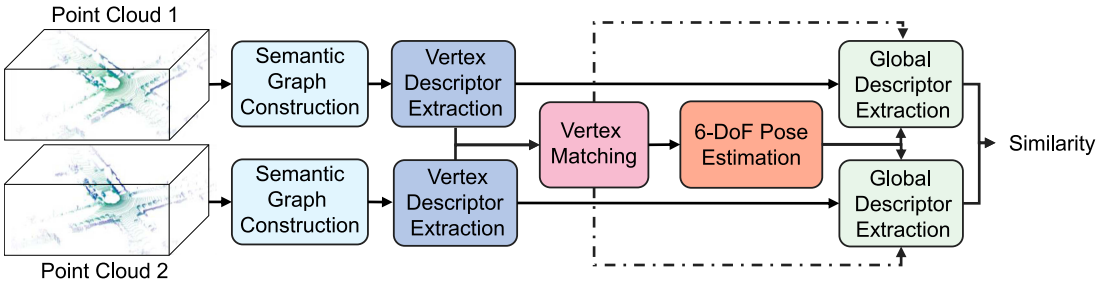


Fig. 1. Framework of our proposed method. Point Cloud 1 and Point Cloud 2 respectively refer to an off-line point cloud from a pre-built database and an on-line point cloud captured at the current moment from a vehicle-mounted LiDAR.

filtering methods require a sequence of on-line point clouds from different moments for filter algorithms, such as Bayesian filter. In this review, we only focus on one-shot methods, and divide them into non-6-DoF and 6-DoF methods.

A. Non-6-DoF Global Localization

In this letter, non-6-DoF includes 1-DoF and 3-DoF. The 1-DoF global localization only estimates headings (i.e., yaw angles), while the 3-DoF localization estimates 2-D positions (i.e., x and y coordinates) and headings.

Existing methods usually convert point clouds from 3-D to 2-D, such as bird-eye-view (BEV) images and range images, to extract their descriptors. Scan Context [1] is a typical BEV-based method, in which raw point clouds are converted into BEV polar grids. The largest z coordinate among the points in each block of the polar grids is used to build a matrix-based global descriptor. The heading difference between the query frame and the matching frame is estimated by column shifting when their similarity reaches the maximum. Based on Scan Context, Wang et al. [2] combined intensity information with geometric information for descriptor extraction. Only 1-DoF pose estimation is provided in [1] and [2]. To achieve 3-DoF global localization, Wang et al. [3] proposed a global descriptor, LiDAR-Iris, which is extracted from the generated binary signature image of the LiDAR point cloud. The 3-DoF pose estimation is obtained from the Iris after Fourier transform. Xu et al. [4] transformed the Scan Context-based feature into the frequency domain to learn a rotation-invariant descriptor and pose. Kim et al. [5] proposed Scan Context++ by extending Scan Context with a Cartesian BEV-based descriptor to obtain the translation estimation. Instead of using geometric information, Li et al. [6] explored semantic information to extract Scan Context-based global descriptor. A two-step global semantic Iterative Closest Point method was proposed to find the 3-DoF pose, which was further used to improve matching performance. Instead of using polar grids, Ding et al. [7] performed Radon Transform to extract descriptors based on BEV images, which can estimate headings between point clouds without being affected by the translation variations. Lu et al. [8] also used Radon Transform for descriptor extraction, while the proposed learning-free descriptor can estimate both headings and translations.

Differently, some researchers used a spherical projection model to convert point clouds into range images instead of 2-D BEV images. Chen et al. [9] proposed OverlapNet, a deep neural network that can estimate overlap and relative yaw angle between two LiDAR point clouds. Lukas et al. [10] proposed a data-driven system for place recognition that estimates yaw discrepancies between LiDAR point-cloud scans at the same time.

B. 6-DoF Global Localization

A common solution to achieve 6-DoF pose estimation relies on the correspondence of geometric elements (e.g., key points, lines, and planes). Shan et al. [11] projected 3-D point clouds to get intensity images, in which ORB feature descriptors are extracted and used for place recognition queries using BoW. The matched candidate is further validated by Perspective-n-Point (PnP) and Random Sample Consensus (RANSAC). Giammarino et al. [12] conducted experiments to show that straightforward adaptation of existing visual place recognition techniques on intensity information can achieve reliable loop closure detection. However, an expensive high-resolution LiDAR is usually needed to obtain dense intensity images for feature extraction.

Recently, some researchers directly extracted 3-D features from point clouds for local matching and global descriptors for place recognition. For example, Cattaneo et al. [13] proposed LCDNet, an end-to-end approach for both place recognition and pose estimation. A PV-RCNN-based network is used to extract local descriptors. The proposed pose regression network can register two point clouds with an arbitrary initial misalignment. Du et al. [14] integrated multi-level context information and channel-wise relations into local features using FlexConv and an SE block. The extracted local features are then aggregated as a global descriptor for place recognition. The 6-DoF pose estimation is also achieved by the local feature matching. Komorowski et al. [15] proposed EgoNN, a deep neural network based on MinkLoc3D to extract both local and global descriptors. Different from DH3D, EgoNN can process more larger point clouds, while the number of points should be less than 10,000 in DH3D. Instead of performing place recognition based on the global descriptor generated from local features, Cui et al. [16] used 3-D features LinK3D [17] to build a BoW for loop closure detection. The full 6-DoF pose estimation was then obtained by using RANSAC and Singular Value Decomposition (SVD) on the point-to-point LinK3D matching results. Boss et al. [18] and Guo et al. [19] both used probabilistic voting strategy for place recognition. The 6-DoF poses are estimated based on 3-D keypoint and intensity-integrated keypoint, respectively.

The above methods all use point-level features, which are not memory efficient in large-scale environments. Instead, some researchers use semantic objects to build descriptors.

Authorized licensed use limited to: CITY UNIV OF HONG KONG. Downloaded on February 28,2024 at 07:01:23 UTC from IEEE Xplore. Restrictions apply.

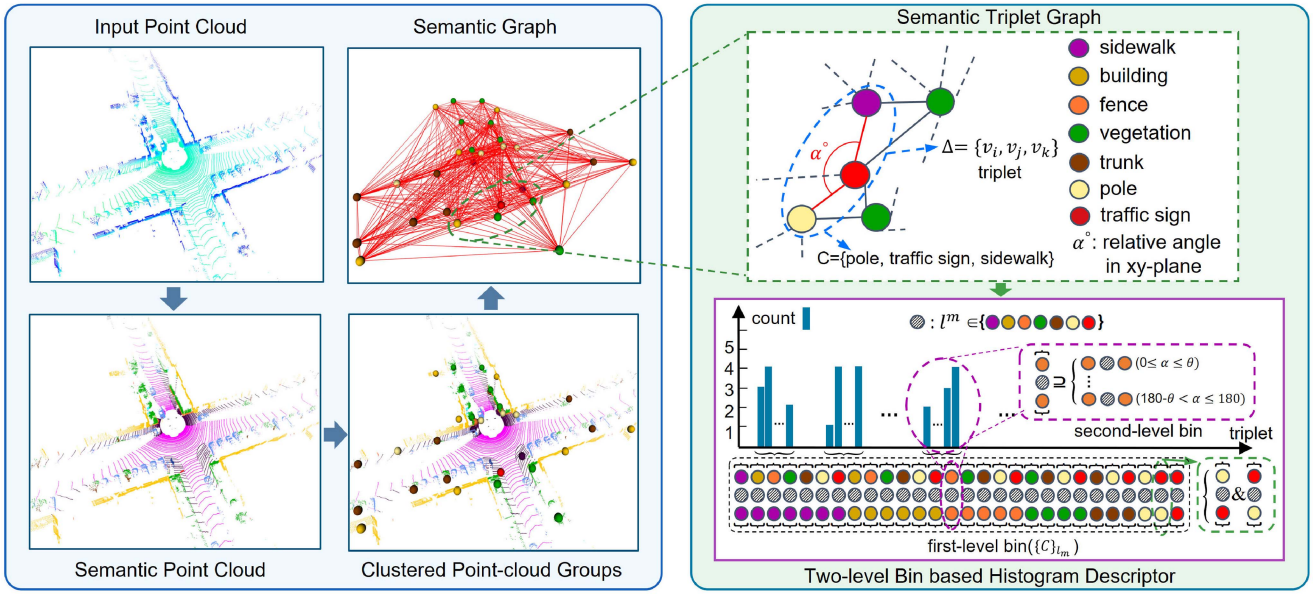


Fig. 2. (a) Semantic graph construction and (b) vertex descriptor extraction. Semantic segmentation is first performed on the input LiDAR point cloud. A semantic graph is then constructed based on the clustered objects. The blue dotted ellipse shows a sample triplet. Finally, the triplet semantic histogram-based descriptor is employed to represent the vertices in the graph. l_m could be an arbitrary label from set L.

GOSMatch [20] represents environments as semantic object graphs. Histogram-based descriptor was proposed for both global descriptor and vertex descriptor. Similarly, BoxGraph [21] builds a semantic object fully-connected graph from point clouds. The authors combined the object shape and centroid information into the vertex descriptor for both place recognition and pose estimation.

Like GOSMatch and BoxGraph, our method also uses a semantic graph to encode environment information. But differently, we explicitly use geometric, semantic, and topological information from stable static objects in the environment. GOSMatch [20] uses vehicle information, which is not stable and consistent when time changes. BoxGraph [21] does not explicitly use topological information for descriptor extraction. We believe that the topological information is beneficial for building a more descriptive representation for a semantic graph.

III. The Proposed Method

The framework of our proposed method is shown in Fig. 1. The method is generally divided into four parts: semantic-graph construction, vertex descriptor extraction and matching, 6-DoF pose estimation, as well as global descriptor extraction and similarity computing.

A. Semantic Graph Construction

Fig. 2(a) shows the semantic graph construction process. Given a point cloud, we first find its semantic segmentation result. We only consider static objects in this work. The class label is denoted as l, in which $l \in L = \{$sidewalk, building, fence, vegetation, trunk, pole, and traffic sign$\}$. Points with the same class label are then clustered into different object groups using Euclidean Cluster algorithm implemented by Point Cloud Library [22]. Each clustered group has its own ID, class label, and corresponding points. To improve the consistency of object clustering across different frames, we treat the clustered group as one individual object only when the number of points within the group is larger than a pre-defined threshold, which varies with the class label of the group. It is larger for sidewalk, building, and vegetation than the other classes.

Similar to [20], [21], we represent the clustered point-cloud groups as an undirected graph $\boldsymbol{G} = \langle \boldsymbol{V}, \boldsymbol{E} \rangle$. $\boldsymbol{V} = \{v_i\}$ is the set of vertices, in which v_i refers to an individual object-i in the point cloud. Each vertex contains the geometric centroid position (i.e., xyz-coordinate) and class label-l of the object. $\boldsymbol{E} = \{e_{ij}\}$ is the set of edges that connect two different vertices, in which $e_{ij} = \langle v_i, v_j \rangle$ is the edge connecting vertices v_i and v_j. Two objects are connected only when their distance is less than a pre-defined threshold $\tau_{edge} = 55$ m. As shown in Fig. 3, the semantic graph becomes a completely connected graph when τ_{edge} is large enough.

B. Triplet-Based Vertex Descriptor Extraction and Matching

The geometric centroids of the clustered objects can be regarded as an abstract representation of the point cloud at the instance level. To find the relative pose between two scans based on two semantic graphs, vertices correspondences are needed. Inspired by the semantic histogram-based descriptor used in [23], we also use a semantic histogram descriptor to embed semantic and topological information for vertices in a graph. Differently, we explicitly embed geometric information into the histogram based on the relative angle among vertices. To calculate such a relative angle, at least three vertices are needed. However, using more vertices (e.g., >3) to calculate the relative angle at once can lead to higher computational complexity. So, we only use three vertices and use the word *triplet* to name our descriptors.

Authorized licensed use limited to: CITY UNIV OF HONG KONG. Downloaded on February 28,2024 at 07:01:23 UTC from IEEE Xplore. Restrictions apply.

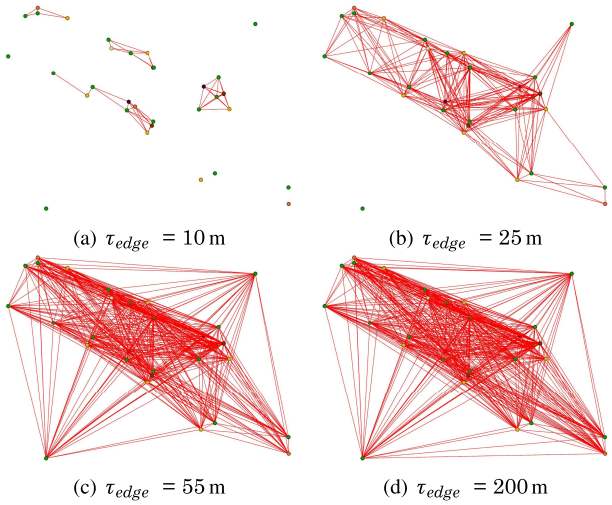


Fig. 3. Semantic graphs with different values of τ_{edge}.

We denote $\Delta = \{v^f, v^m, v^t\}$ as a triplet, in which f, m, and t are short for $first$, $middle$, and $third$, respectively. Δ_{ijk} refers to the triplet when $v^f = v_i$, $v^m = v_j$, $v^t = v_k$, in which $i \neq j \neq k$. v_i and v_k are both connected to v_j. The relative angle of Δ in the xy-plane is denoted as α. $C = \{l^f, l^m, l^t\}$ denotes the class combination for a triplet. An example of a Δ can be found in the blue dotted ellipse at the top right of Fig. 2(b). Note that Δ_{ijk} and Δ_{kji} are regarded as the same Δ because objects in point clouds are unordered. $\{\Delta\}_{v_j}$ is the set of all the triplets that satisfy $v^m = v_j$ in the graph.

1) Histogram Based on Two-Level Bins: To build a histogram, the first step is to "bin" the range of values, that is, to divide the entire range of values into a set of intervals. We innovatively design two-level bins to classify and count triplets.

We design the first-level bins based on class combinations of triplets. Specifically, given the class label set L, we first chose one class label for l^m of C, such as l^m = trunk. Then, we exhaustively chose class labels for l^f and l^t. All these class combinations form the first-level bins of the histogram for vertex whose class label is trunk, denoted as $\{C\}_{l^m=\text{trunk}}$. Since objects in point clouds are unordered, we regard {fence, trunk, pole} and {pole, trunk, fence} as the same C. The length of $\{C\}_{l^m=\text{trunk}}$ can be calculated by $N_1 = \sum_{w=1}^{|L|} \mathbf{C}_w^1$, in which $|L|$ is the cardinality of L and $\mathbf{C}$ refers to Combination in mathematics ($|L| = 7$ in this work, so $N_1 = 28$). Similarly, the first-level bins of the histogram for vertex with the other class labels can be obtained by the same way, such as $\{C\}_{l^m=\text{pole}}$, etc. It is worth noting that the first-level bins $\{C\}_{l^m}$ are different for vertices whose class labels are different, as shown in the bottom right of Fig. 2(b).

For each first-level bin $C \in \{C\}_{l_m}$, we further divide it into N_2 second-level bins. Each second-level bin has the same combination as C, but with a different relative angle range. As shown in the purple dotted box in Fig. 2(b), we have $N_2 = 180°/\theta$, in which θ (we set θ as 5°) is the interval of the relative angle range. Note that θ should be divisible by 180°. When $\theta = 180°$, the second-level bins disappear, which means that only semantic combinations of triplets are embedded in the histograms, leading to performance degradation. The details can be found in our supplementary material, which is available at our GitHub page.

2) Vertex Descriptor Extraction: Given a vertex v_j from a constructed graph, we first build the first-level bins (i.e., $\{C\}_{l^m}$) for its vertex histogram descriptor according to the class label of v_j. Then, we search the graph to get $\{\Delta\}_{v_j}$. For each Δ in $\{\Delta\}_{v_j}$, we find the corresponding first-level bin that has the same class combination C as Δ. By comparing α with the cover range of relative angles for the second-level bins, we can easily know which second-level bin the current triplet Δ belongs to. Then, the triplet number in the retrieved second-level bin is increased by one. Repeat the above steps for all the Δ in $\{\Delta\}_{v_j}$, we can obtain the final triplet-based histogram descriptor for v_j, see Fig. 2(b). The extracted vertex descriptor can be easily converted to a $N_1 \times N_2$ matrix, denoted as Des_{v_j}. Each element in the matrix equals the number of triplets in $\{\Delta\}_{v_j}$ with the same class combination C and range of α in the graph.

3) Vertex Matching: Vertex matching between two graphs $\boldsymbol{G_1}$ and $\boldsymbol{G_2}$ can be realized by comparing the similarity between the descriptors of vertices in the two graphs. Only vertices with the same class label are compared, for example, trunks in $\boldsymbol{G_1}$ will only be compared with trunks in $\boldsymbol{G_2}$. We use the cosine similarity to compare the similarity:

$$\text{Sim}\left(v_j^1, v_t^2\right) = \frac{\sum Des_{v_j^1} \cdot Des_{v_t^2}}{\left\|Des_{v_j^1}\right\|_F \times \left\|Des_{v_t^2}\right\|_F}, \tag{1}$$

in which Sim is short for similarity, the dot $\cdot$ is the element-wise multiplication on two matrices, $\|\cdot\|_F$ is the Frobenious norm of a matrix, $\sum$ is the summation of all elements of a matrix, v_j^1 and v_t^2 are respectively arbitrary vertices in $\boldsymbol{G_1}$ and $\boldsymbol{G_2}$ with the same class label. For every v_j^1 in $\boldsymbol{G_1}$, we chose the vertex v_t^2 in $\boldsymbol{G_2}$ that has the highest similarity between v_j^1 as the matching result, denoted as (v_j^1, v_t^2). We use $\{(v_j^1, v_t^2)\}$ to represent all vertex matching results between $\boldsymbol{G_1}$ and $\boldsymbol{G_2}$.

C. 6-DoF Pose Estimation

There are mainly two steps in the 6-DoF pose estimation. Firstly, we use RANSAC [24] and SVD to estimate a coarse pose, $\widetilde{\mathrm{T}} \in SE(3)$, based on the vertex matching results $\{(v_j^1, v_t^2)\}$. Secondly, we optimize the pose estimation by minimizing the total projection error based on the inliers matches after the first step. For two given matched vertices, the projection error can be obtained as $\varepsilon = \|\bar{v}_j^1 - T \cdot \bar{v}_t^2\|$, in which $T \in SE(3)$ is the pose needed to be optimized, $\bar{v}_j^1$ and $\bar{v}_t^2$ are respectively the homogeneous coordinates for the vertices v_j^1 and v_t^2. The Ceres Solver [25] is used to solve this optimization problem. We denote the optimal 6-DoF pose obtained through the optimization process as T^*. To speed up the optimization process, we use $\widetilde{\mathrm{T}}$ as the initial guess for the optimization.

D. Global Descriptor Extraction and Similarity Computing

To achieve place recognition, the similarity between two point clouds is required. We aggregate the local (vertex) descriptors into a global descriptor to measure the similarity.

1) Global Descriptor Extraction: Let $\{Des_{v_j}\}$ denote the set of all vertex descriptors in a semantic graph. As mentioned in Section III-B, the first- and second-level bins of the histogram are exactly the same for the vertices whose class labels are the same. Therefore, Des_{v_j} and $Des_{v_{j+1}}$ from the same graph can be directly added together element-wisely when the class labels

Authorized licensed use limited to: CITY UNIV OF HONG KONG. Downloaded on February 28,2024 at 07:01:23 UTC from IEEE Xplore. Restrictions apply.

TABLE I
MAXIMUM F_1 SCORE (%) AND EXTENDED PRECISION (EP) (%) ON THE KITTI DATASET

Methods	Seq-00		Seq-02		Seq-05		Seq-06		Seq-07		Seq-08	
	F_1	EP	F_1	EP	F_1	EP	F_1	EP	F_1	EP	F_1	EP
IRIS (ICRA 2020) [3]	66.8	62.6	76.2	66.6	76.8	74.7	91.3	79.1	62.9	65.1	47.8	56.2
M2DP (IROS 2016) [30]	70.8	61.6	71.7	60.3	60.2	61.1	78.7	68.1	56.0	58.6	7.3	50.0
OLN (AURO 2021) [9]	86.9	55.5	82.7	63.9	92.4	79.6	93.0	74.4	81.8	58.6	37.4	50.0
PNV (CVPR 2018) [31]	77.9	64.1	72.7	69.1	54.1	53.6	85.2	76.7	63.1	59.1	3.7	50.0
SGPR (IROS 2020) [27]	82.0	50.0	75.1	50.0	75.1	53.1	65.5	50.0	86.8	72.1	75.0	52.0
SC (IROS 2018) [1]	75.0	60.9	78.2	63.2	89.5	79.7	96.8	92.4	66.2	55.4	60.7	56.9
ISC (ICRA 2020) [2]	65.7	62.7	70.5	61.3	77.1	72.7	84.2	81.6	63.6	63.8	40.8	54.3
SSC-RN (IROS 2021) [6]	93.9	82.6	89.0	74.5	94.1	90.0	**98.6**	**97.3**	87.0	77.3	88.1	73.2
Ours-RN	99.6	92.6	83.1	**76.4**	98.6	87.8	97.6	86.4	97.8	96.0	96.3	93.2
SSC-SK (IROS 2021) [6]	95.1	84.9	**89.1**	74.8	95.1	90.3	98.5	96.9	87.5	80.5	94.0	93.2
Ours-SK	**99.8**	**99.5**	71.6	74.1	**99.1**	**92.5**	97.1	92.6	**98.1**	**97.9**	**99.0**	**97.3**

Larger values indicate better performance. The best results are highlighted in bold font.

for v_j and v_{j+1} are the same. Instead of directly aggregating all the vertices in a graph into a global descriptor, we use remaining matched vertices. Specifically, we perform the projection operation (as mentioned in Part C) using T^* and the vertex matching results $\{(v_j^1, v_t^2)\}$. Only when the projection error is less than a pre-defined threshold τ_{proj}, the corresponding vertex match is kept. The remaining vertices matches are denoted as $\{(v_j^1, v_t^2)\}_{rem}$. For $\boldsymbol{G_1}$, we add up descriptors for vertices with the same class label-l from $\{(v_j^1, v_t^2)\}_{rem}$, denoted as $Des^{l,1}$. A $Des^{l,1}$ records all the triplets whose middle class is label-l from the remaining vertices in $\boldsymbol{G_1}$, which is the overall description for class-l in $\boldsymbol{G_1}$. Repeat the adding operation for every class label-l. Then, the global descriptor for $\boldsymbol{G_1}$ is the set of all $Des^{l,1}$, denoted as $\{Des^{l,1}\}$, in which $l \in L$. The global descriptor $\{Des^{l,2}\}$ for $\boldsymbol{G_2}$ can be obtained by the same way.

2) Similarity Computing: We use cosine similarity to measure how close two global descriptors are. Specifically, given two global descriptors $\{Des^{l,1}\}$ and $\{Des^{l,2}\}$, we first calculate the similarity between $Des^{l,1}$ and $Des^{l,2}$ for each class-l. Then, a weighted model is adopted to combine the similarity scores for all the classes to get the final score between the global descriptors from the two graphs:

$$\mathrm{Sim}\left(\boldsymbol{G_1}, \boldsymbol{G_2}\right) = \sum_{l \in L} \frac{1}{|L|} \times \frac{\sum Des^{l,1} \cdot Des^{l,2}}{\left\|Des^{l,1}\right\|_F \times \left\|Des^{l,2}\right\|_F}. \quad (2)$$

IV. EXPERIMENTAL RESULTS AND DISCUSSIONS

A. Dataset and Experimental Setup

We evaluate our method using the KITTI odometry dataset [26]. There are 11 sequences with the 6-DoF trajectory ground truth in the dataset. Loop closure only occurs in sequence 00, 02, 05, 06, 07, and 08. So, we only perform experiments on these sequences. Note that sequence-08 contains reverse loops (i.e., revisiting the same place from the opposite direction), while loop closure only happens with the same direction in the other sequences. To ensure a fair comparison, we follow the same setup as in [27] and [6]. They use a large number of point-cloud pairs for evaluation. A true-positive loop closure occurs when the relative distance is less than 3 m between a point cloud pair. If the distance exceeds 20 m, the corresponding pair is considered as a negative sample. We use the benchmark and evaluation pairs provided by [6]. For each sequence, there are N_p positive samples and $100 \cdot N_p$ negative samples that are selected randomly.

Following [6], we use the ground-truth semantic labels from the SemanticKITTI dataset [28], and the prediction labels from RangeNet++ [29] as the point-cloud semantic segmentation results, respectively denoted as SK and RN in Tables I and II. Due to the page limit, we present the parameter tuning results for τ_{edge}, θ, and τ_{proj} in terms of both place recognition and pose estimation in our supplementary material.

B. Place Recognition Performance

We compare our method with the open-sourced state-of-the-art methods, including Lidar Iris (IRIS) [3], M2DP [30], Overlapnet (OLN) [9], PointNetvlad (PNV) [31], SGPR [27], Scan Context (SC) [1], Intensity Scan Context (ISC) [2], and SSC [6]. In this letter, we directly import the results from [6] for all the above methods.

To ensure comprehensive evaluation, we not only show the Precision-Recall curves (see Fig. 4), but also provide the maximum F_1 score and the Extended Precision (EP) [32] as shown in Table I. The F_1 score is calculated as $F_1 = 2\frac{PR}{P+R}$, in which P is precision and R is recall. For EP, we have $EP = \frac{1}{2}(P_{R0} + R_{P_{100}})$, in which P_{R0} is the precision at the minimum recall, and $R_{P_{100}}$ equals to the maximum recall at 100% precision.

As we can see in Fig. 4 and Table I, our approach outperforms the other methods except on Seq-02 and Seq-06. SSC [6] outperforms our method slightly on Seq-02 and 06. We conjecture the reason could be the degradation of object clustering or inconsistency of objects between two LiDAR point clouds (i.e., some objects can not be both observed in two scans). When there are too few objects in the scene, the constructed semantic graph would be very sparse, leading to degradation in the descriptive capability of the vertex descriptor and global descriptor. Especially on Seq-02, we found that the average number of clustered objects is obviously less than that of the other sequences, which is shown as the yellow bars in Fig. 5. As for Seq-08, there exist many reverse loop closures (opposite viewpoints), our method can still achieve satisfactory performance of place recognition. This demonstrates the robustness of our method against viewpoint changes.

Interestingly, the place recognition performance between Ours-SK and Ours-RN is very close. This indicates that our method can be practically applicable using standard semantic segmentation networks. We guess the reason is that our method relies more on the consistency of semantic segmentation (i.e.,

Authorized licensed use limited to: CITY UNIV OF HONG KONG. Downloaded on February 28,2024 at 07:01:23 UTC from IEEE Xplore. Restrictions apply.

TABLE II
AVERAGE RTE (M) AND AVERAGE RRE (DEGREE) WITH STANDARD DEVIATIONS ON THE KITTI DATASET

		Seq-00	Seq-02	Seq-05	Seq-06	Seq-07	Seq-08
RTE	SSC-RN	0.38±0.43	1.07±0.81	0.51±0.59	0.33±0.37	0.35±0.45	0.45±0.57
	Ours-RN	**0.07±0.05**	0.95±1.61	0.16±1.01	**0.07±0.05**	0.35±1.59	0.55±2.23
	SSC-SK	0.32±0.41	**0.91±0.81**	0.47±0.60	0.30±0.38	0.30±0.33	0.25±0.33
	Ours-SK	**0.07±0.13**	1.66±3.72	**0.11±0.56**	0.09±0.10	**0.22±0.10**	**0.20±1.09**
RRE	SSC-RN	0.58±2.24	**0.87±1.15**	0.66±4.53	0.64±0.81	1.43±10.43	1.35±1.37
	Ours-RN	0.37±0.28	4.52±20.69	1.05±9.64	**0.26±0.16**	2.13±14.46	2.34±9.87
	SSC-SK	0.80±5.75	1.22±6.03	0.62±0.67	0.76±0.79	**0.52±2.16**	1.55±1.37
	Ours-SK	**0.34±0.31**	16.74±44.56	**0.58±6.30**	0.27±0.27	1.75±13.08	**1.04±2.56**

Smaller values indicate better performance. The best results are highlighted in bold font.

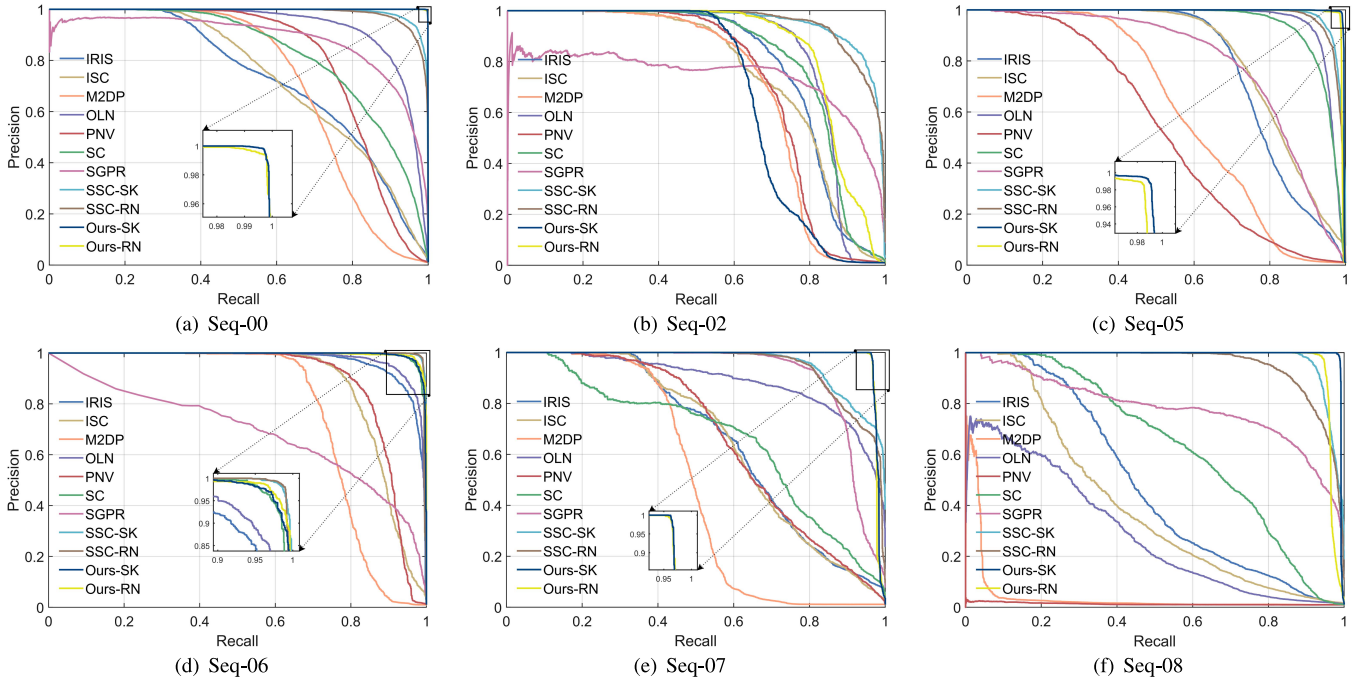


Fig. 4. Precision-Recall curves on the KITTI dataset. Seq is short for Sequence. RN and SK represent prediction labels from RangeNet++ and ground-truth semantic labels from SemanticKITTI dataset, respectively.

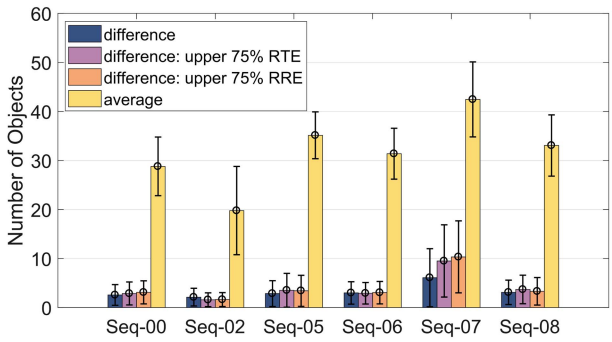


Fig. 5. Distribution of the numbers of clustered objects for different KITTI sequences in the form of bar chart. Results are obtained based on the ground-truth semantic labels. Seq is short for Sequence.

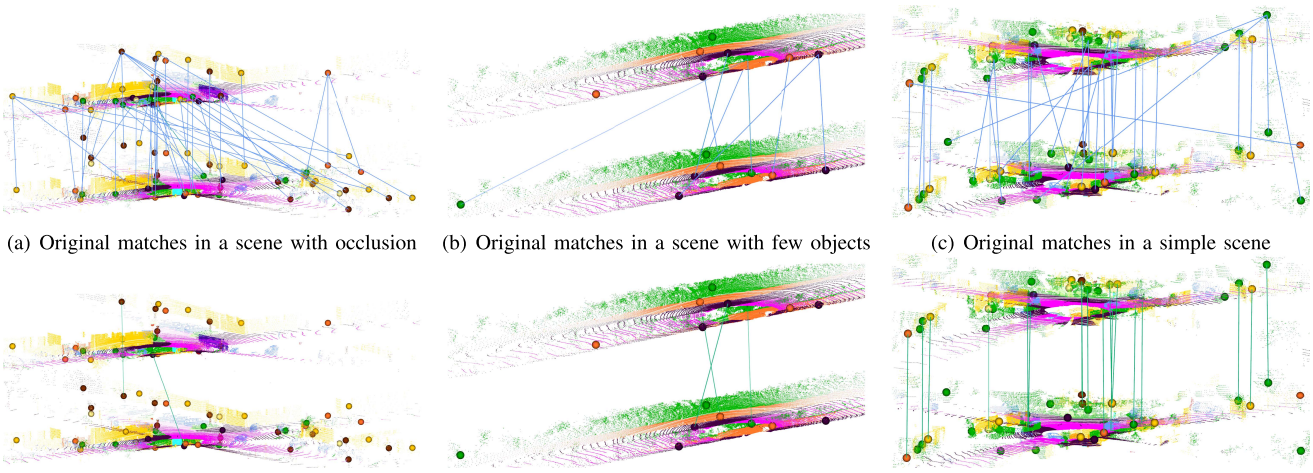
similar segmentation results for different observations around the same location) rather than higher segmentation accuracy.

C. Pose Estimation Performance

We compare our results with SSC [6] on the positive pairs. Note that SSC can only provide 3-DoF pose estimation (x, y, yaw), while ours provides 6-DoF pose estimation. Relative Translation Error (RTE) and Relative Rotation Error (RRE) [33] are employed to evaluate the translation and orientation accuracy, respectively. RTE is calculated as $RTE = \|t_{est} - t_{gt}\|_2$, in which $t_{est} \in T^*$ is the estimated translation, t_{gt} is the ground-truth for the estimated translation. RRE is calculated as $RRE = \cos^{-1}\left(\frac{\mathrm{Tr}(R_{gt}^{-1}R_{est})-1}{2}\right)$ or $RRE = \cos^{-1}\left(\frac{\mathrm{Tr}(R_{est}^{T}R_{gt})-1}{2}\right)$, in which $R_{est} \in T^*$ is the estimated rotation matrix, R_{gt} is the ground truth for the estimated rotation matrix, and $Tr(\cdot)$ represents the trace operation.

Table II shows the RTE average and RRE average with standard deviations. We can see that our method presents better translation estimations than SSC on almost all the sequences except Seq-02. Especially on Seq-00 and Seq-06, the average RTE of our method is less than 10 cm. As for orientation estimation, SSC only outperforms ours on Seq-02 and Seq-07.

Authorized licensed use limited to: CITY UNIV OF HONG KONG. Downloaded on February 28,2024 at 07:01:23 UTC from IEEE Xplore. Restrictions apply.



(a) Original matches in a scene with occlusion (b) Original matches in a scene with few objects (c) Original matches in a simple scene

(d) Remaining matches in a scene with occlusion (e) Remaining matches in a scene with few objects (f) Remaining matches in a simple scene

Fig. 6. Examples for original vertices matches (the first row) and the remaining vertices matches (the second row). Original vertices matches represent the matching results from Section III-B. Remaining vertices matches refer to the selected vertices for global descriptor extraction in Section III-D. In each figure, the upper and lower point cloud represents the visited and the revisiting frame, respectively. The figure is best viewed in color.

On the other sequences, our method outperforms SSC with a large margin, which can generally achieve an average RRE at the sub-degree level. In general, our method can provide a competitive metric-level 6-DoF pose estimation against SSC (only 3-DoF available).

As we can see in Table II, on Seq-02, our method cannot well estimate the poses. There are two main possible reasons for this: 1) There might be too many incorrect vertices matches, leading to inferior optimization results; 2) Too few vertices matches are obtained, which cannot provide enough constraints during the optimization process. To better assess the difference between clustered objects in different sequences, we calculate the average number of the clustered objects (i.e., the average number of objects in an input LiDAR point-cloud pair). In addition, object number difference (i.e., the absolute value of object number difference between the LiDAR point-cloud pair) and the object number difference among pairs whose RTE/RRE is larger than the upper quartile of RTE/RRE are also calculated.

The average values and standard deviations of the above items for every sequence are visualized in Fig. 5. Generally, small object number difference refers to better consistency of clustered objects in two frames, which should usually have a better vertex matching result. Besides, a larger average object number contributes to a denser graph, which can also result in a better vertex matching result. Although the object number difference is small on Seq-02, we find that its average object number is the smallest, which is less than 20, while the other sequences have at least 30 objects for each scene on average. In addition, we find that all the three object number differences on Seq-07 are much larger than those on the other sequences. Such a huge difference leads to degradation in pose estimation even Seq-07 has the largest average object number. These results are all consistent with our conjecture. Some examples of vertices matches are visualized, including a failing example with occlusion caused by a truck (see Fig. 6(a) and (d)), a failing example with few clustered objects (see Fig. 6(b) and (e)), and a success example (see Fig. 6(c) and (f)). The degradation of vertex matching can lead to the failure of pose estimation, which would further affect place recognition. In conclusion, a larger average object number and less object number difference can significantly improve the pose estimation performance.

TABLE III
MAXIMUM F_1 SCORE (%) AND EXTENDED PRECISION (EP) (%) ON THE KITTI DATASET WITH/WITHOUT VERTICES MATCHES SELECTION OPERATION FOR GLOBAL DESCRIPTOR EXTRACTION

Sequence	F_1		EP	
	(w/ select)	(w/o select)	(w/ select)	(w/o select)
Seq-00	99.8	57.3	99.5	56.9
Seq-02	71.6	17.9	74.1	50.4
Seq-05	99.1	56.7	92.5	50.9
Seq-06	97.1	73.1	92.6	64.0
Seq-07	98.1	76.6	97.9	73.6
Seq-08	99.0	47.7	97.3	53.0
average	94.1	54.9	92.3	58.1

D. Ablation Study

To demonstrate the contribution of the projection-based selection operation, we disable the selection operation and directly generate the global descriptor using all vertices. Note that our ablation study is conducted using ground-truth semantic labels to avoid the impact caused by the quality of semantic segmentation. Results can be found in Table III. As we can see, the vertices matches selection operation can significantly improve the performance of place recognition in terms of both maximum F_1 score and EP. The average maximum F_1 score and average EP are increased by 0.392 and 0.342, respectively.

E. Memory Consumption

Compared to using raw point clouds, only vertex descriptors need to be accessed in our method. On average, throughout all the sequences, there are 31 clustered objects in a graph. So, the average memory consumption for a graph is $(N_1 \times N_2 + 4) \times 4 \times 31$ bytes. In our case, this equates to 125.5 KB. The average number of 3-D points per point-cloud scan is 123,584, which approximately consumes 1483.0 KB of memories. This demonstrates our high efficiency in terms of memory consumption.

Authorized licensed use limited to: CITY UNIV OF HONG KONG. Downloaded on February 28,2024 at 07:01:23 UTC from IEEE Xplore. Restrictions apply.

V. Conclusions and Future Work

We presented here a novel method, Triplet-Graph, to achieve global metric localization with a single on-line point cloud obtained at the current moment. For every input point cloud, a semantic graph is first constructed to represent a scene compactly. We design a triplet-based semantic histogram descriptor to encode both geometric information and topological information for every vertex in the semantic graph. These descriptors are then used to perform pose estimation and are aggregated into a global descriptor for place recognition. We presented both quantitative and qualitative results for our method, which suggest that our method generally outperforms the state-of-the-art methods. However, our method still has limitations. For example, it suffers from inconsistencies of clustered objects from different observations. When there are not enough objects in a scene, or the number of objects varies greatly during the on-line and off-line stages, our method could perform poorly. In the future, we will combine visual cues instead of merely using point clouds to improve our method.

References

[1] G. Kim and A. Kim, "Scan Context: Egocentric spatial descriptor for place recognition within 3D point cloud map," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 4802–4809.

[2] H. Wang, C. Wang, and L. Xie, "Intensity scan context: Coding intensity and geometry relations for loop closure detection," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2020, pp. 2095–2101.

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

[4] X. Xu, H. Yin, Z. Chen, Y. Li, Y. Wang, and R. Xiong, "DISCO: Differentiable scan context with orientation," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 2791–2798, Apr. 2021.

[5] G. Kim, S. Choi, and A. Kim, "Scan Context++: Structural place recognition robust to rotation and lateral variations in urban environments," *IEEE Trans. Robot.*, vol. 38, no. 3, pp. 1856–1874, Jun. 2022.

[6] L. Li et al., "SSC: Semantic scan context for large-scale place recognition," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 2092–2099.

[7] X. Ding et al., "Translation invariant global estimation of heading angle using sinogram of LiDAR point cloud," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2022, pp. 2207–2214.

[8] S. Lu, X. Xu, H. Yin, Z. Chen, R. Xiong, and Y. Wang, "One ring to rule them all: Radon sinogram for place recognition, orientation and translation estimation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 2778–2785.

[9] X. Chen, T. Läbe, A. Milioto, T. Röhling, J. Behley, and C. Stachniss, "OverlapNet: A siamese network for computing LiDAR scan similarity with applications to loop closing and localization," *Auton. Robots*, vol. 46, pp. 61–81, 2021.

[10] L. Schaupp, M. Bürki, R. Dubé, R. Siegwart, and C. Cadena, "OREOS: Oriented recognition of 3D point clouds in outdoor scenarios," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 3255–3261.

[11] T. Shan, B. Englot, F. Duarte, C. Ratti, and D. Rus, "Robust place recognition using an imaging LiDAR," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2021, pp. 5469–5475.

[12] L. Di Giammarino, I. Aloise, C. Stachniss, and G. Grisetti, "Visual place recognition using LiDAR intensity information," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 4382–4389.

[13] D. Cattaneo, M. Vaghi, and A. Valada, "LCDNet: Deep loop closure detection and point cloud registration for LiDAR SLAM," *IEEE Trans. Robot.*, vol. 38, no. 4, pp. 2074–2093, Aug. 2022.

[14] J. Du, R. Wang, and D. Cremers, "DH3D: Deep hierarchical 3D descriptors for robust large-scale 6DoF relocalization," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 744–762.

[15] J. Komorowski, M. Wysoczanska, and T. Trzcinski, "EgoNN: Egocentric neural network for point cloud based 6DoF relocalization at the city scale," *IEEE Robot. Automat. Lett*, vol. 7, no. 2, pp. 722–729, Apr. 2022.

[16] 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. Automat. Lett*, vol. 8, no. 5, pp. 2828–2835, May 2023.

[17] Y. Cui, Y. Zhang, J. Dong, H. Sun, and F. Zhu, "Link3D: Linear keypoints representation for 3D LiDAR point cloud," *IEEE Robot. Automat. Lett.*, 2024.

[18] M. Bosse and R. Zlot, "Place recognition using keypoint voting in large 3D LiDAR datasets," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2013, pp. 2677–2684.

[19] J. Guo, P. V. Borges, C. Park, and A. Gawel, "Local descriptor for robust place recognition using LiDAR intensity," *IEEE Robot. Automat. Lett*, vol. 4, no. 2, pp. 1470–1477, Apr. 2019.

[20] Y. Zhu, Y. Ma, L. Chen, C. Liu, M. Ye, and L. Li, "Gosmatch: Graph-of-semantics matching for detecting loop closures in 3D LiDAR data," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 5151–5157.

[21] G. Pramatarov, D. De Martini, M. Gadd, and P. Newman, "Boxgraph: Semantic place recognition and pose estimation from 3D LiDAR," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 7004–7011.

[22] R. B. Rusu and S. Cousins, "3D is here: Point cloud library (PCL)," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2011, pp. 1–4.

[23] X. Guo, J. Hu, J. Chen, F. Deng, and T. L. Lam, "Semantic histogram based graph matching for real-time multi-robot global localization in large scale environment," *IEEE Robot. Automat. Lett*, vol. 6, no. 4, pp. 8349–8356, Oct. 2021.

[24] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[25] S. Agarwal, K. Mierle, and T. C. S. Team, "Ceres solver," 2023. [Online]. Available: https://github.com/ceres-solver/ceres-solver

[26] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The kitti vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 3354–3361.

[27] X. Kong et al., "Semantic graph based place recognition for 3D point clouds," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 8216–8223.

[28] J. Behley et al., "Semantickitti: A dataset for semantic scene understanding of LiDAR sequences," in *Proc. IEEE Int. Conf. Comp. Vis.*, 2019, pp. 9297–9307.

[29] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, "RangeNet++: Fast and accurate LiDAR semantic segmentation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 4213–4220.

[30] 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.*, 2016, pp. 231–237.

[31] M. A. Uy and G. H. Lee, "PointNetVLAD: Deep point cloud based retrieval for large-scale place recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4470–4479.

[32] B. Ferrarini, M. Waheed, S. Waheed, S. Ehsan, M. J. Milford, and K. D. McDonald-Maier, "Exploring performance bounds of visual place recognition using extended precision," *IEEE Robot. Automat. Lett*, vol. 5, no. 2, pp. 1688–1695, Apr. 2020.

[33] C. Choy, J. Park, and V. Koltun, "Fully convolutional geometric features," in *Proc. IEEE Int. Conf. Comp. Vis.*, 2019, pp. 8958–8966.

Authorized licensed use limited to: CITY UNIV OF HONG KONG. Downloaded on February 28,2024 at 07:01:23 UTC from IEEE Xplore. Restrictions apply.