# FusionVLAD: A Multi-View Deep Fusion Networks for Viewpoint-Free 3D Place Recognition

Peng Yin, Lingyun Xu, Ji Zhang, and Howie Choset

***Abstract*—Real-time 3D place recognition is a crucial technology to recover from localization failure in applications like autonomous driving, last-mile delivery, and service robots. However, it is challenging for 3D place retrieval methods to be accurate, efficient, and robust to the variant viewpoints differences. In this letter, we propose FusionVLAD, a fusion-based network that encodes a multi-view representation of sparse 3D point clouds into viewpoint-free global descriptors. The system consists of two parallel branches: a spherical-view branch for orientation-invariant feature extraction, and the top-down view branch for translation-insensitive feature extraction. Furthermore, we design a parallel fusion module to enhance the combination of region-wise feature connection between the two branches. Experiments on two public datasets and two generated datasets show that our method outperforms state-of-the-art with robust place recognition accuracy and efficient inference time. Besides, FusionVLAD requires limited computation resources and makes it extremely suitable for low-cost robots' long-term place recognition task.**

***Index Terms*—Recognition, SLAM, visual learning.**

## I. Introduction

MAPPING and localization are fundamental capabilities for mobile robotics, and researchers have extensively studied this area over the past two decades. In many robot applications such as autonomous vehicles, search-rescue robots, delivery robots, and warehouse automation, localization failures can cause severe problems in navigation and decision making, which requires an accurate and real-time 3D place recognition. However, it remains a challenging task in many real scenarios, especially in complex outdoor and GPS-denied environments. Generally, 3D place recognition is considered as a searching problem to find the most similar place on the map by measuring the similarity of correspondences.

Recent studies on 3D feature learning (PointNet [1], PointNet++ [2]) have brought light to the 3D place recognition task. PointNet-based place recognition methods, such as PointNetVLAD [3], LPDNet [4], PCAN [5], can extract place descriptors directly from the 3D point cloud, which allows them to achieve robust place recognition performance on public datasets. This direct matching is either robust but intractable on the mobile platform, or optimized for efficiency but fragile. The main drawback of PointNet-based methods is that they are sensitive to orientation differences, which are the typical situations in large-scale place recognition tasks. As depicted in Fig. 1, long-term odometry drift will introduce massive orientation differences.

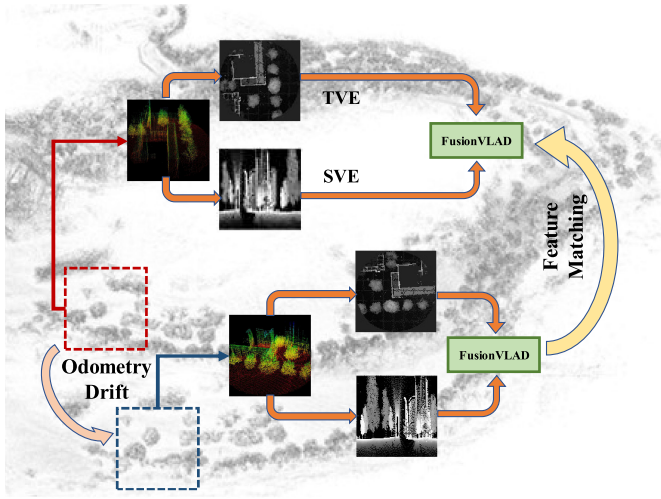


Fig. 1. **Fusion enhanced place recognition.** We generate multi-view projections from 3D point cloud, which are then fed into the proposed FusionVLAD network to extract global descriptors. Different projections (spherical view, top-down view) can capture different perspectives of the original 3D map.

In this letter, we propose FusionVLAD, a parallel fusion network structure to learn the point cloud representations from multi-view projections, and embed them into viewpoint-free low-dimensional place descriptors for efficient global recognition. As depicted in Fig. 1, FusionVLAD includes two parallel branches: the spherical-view encoding (*SVE*) branch and top-down view encoding (*TVE*) branch. As we can see in Fig. 2, the *SVE* branch can encode the spherical projections into an orientation-invariant feature descriptor. In the top-down view perspective, it has a higher tolerance to local translation differences, which benefits our *TVE* branch for translation-insensitive feature extraction. To further enhance the extracted place descriptor inherits orientation- and translation- invariant properties, we design a "parallel fusion" layer between *SVE* and *TVE*, which enables the end-to-end training and inferencing for the global place descriptors.

The key contributions of this letter are summarized as follows:

- We propose a viewpoint-free 3D place recognition system, FusionVLAD. Compared with current state-of-the-art methods, our method show better place recognition ability in most cases under variant viewpoint differences.

Manuscript received September 22, 2020; accepted January 5, 2021. Date of publication February 23, 2021; date of current version March 12, 2021. This letter was recommended for publication by Associate Editor F. Fraundorfer and Editor S. Behnke upon evaluation of the reviewers' comments. *(Corresponding author: Lingyun Xu.)*

The authors are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213 USA (e-mail: pyin2@andrew.cmu.edu; hitmaxtom@gmail.com; zhangji@cmu.edu; choset@cs.cmu.edu).

Digital Object Identifier 10.1109/LRA.2021.3061375

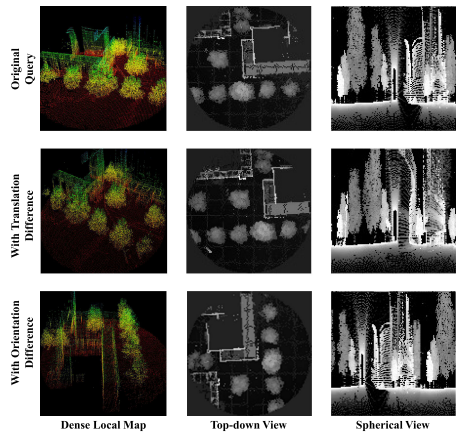


Fig. 2. **Different representations from local dense map.** Second and third row represent the same area under locally translation or orientation difference respectively. Different columns show the dense local map, the top-down and the spherical projections.

- We design a parallel multi-views fusion network, which can simultaneously encode spherical and top-down projections for robust place description learning.
- Based on traditional triplet loss in place recognition, we design a lazy viewpoint-free metric to enhance the learnt descriptors to viewpoint-invariant place recognition.
- We design the experiments with both public and self-recorded datasets to evaluate the place recognition performance of our method under variant translational and rotational differences.

Notably, with public odometry datasets and collected datasets from our hardware platform, the experiment results show that our method is remarkably more robust to viewpoint differences than other methods. Additionally, our FusionVLAD consumes less GPU memory and can inference each place within 18 ms, making it suitable for low-cost robot systems in large-scale mapping applications.

## II. Related Work

Place recognition has been widely applied in SLAM and navigation tasks. In this letter, we focus on 3D LiDAR-based large-scale place recognition. In this section, we will briefly review the relative works for 3D place recognition.

Unlike SIFT [6] and SURF [7] in image fields, handcrafted features for the 3D point-cloud did not achieve great success.

Distance and angle based feature ESF [8], structure based Scan-Context [9], and histogram-based feature SHOT [10] are both focused on local areas and not suitable for extracting global descriptors. Fast Histogram [11] uses a histogram to represent the entire point-cloud distribution. M2DP [12] projects all points to several planes, and they can obtain a global descriptor. These methods are robust to the orientation differences but sensitive to the translation differences.

Traditional LiDAR-based place recognition methods [13] usually rely on a global, off-line, high-resolution map, and can achieve centimeter-level localization, but at the cost of time-consuming off-line map registration and data storage requirements. Recently, learning-based point-cloud descriptors have shown outstanding performance and gradually replacing handcrafted features. Instead of extracting raw LiDAR points in the current scan for geometry matching, SegMatch [14] proposed a matching-by-segmentation method where static objects are segmented out as landmarks to reduce the matching complexity. However, the 3D point-level segmentation process is computational expensive and the assumption of enough static objects is not always a satisfied in the real world. Based on the PointNet-based [1], [2] feature learning techniques, recent 3D place recognition approaches [3]–[5] have made significant progress. In [3], the authors combined the feature extraction ability of PointNet [1] and VLAD-based [15] feature aggregation to obtain translation-invariant 3D place descriptor. Compared with traditional approaches, the extracted features from PointNetVLAD [3] is robust to locally translation differences. Based on Mikaela's work, LPDNet [4] further improved place recognition accuracy by enhancing the local feature extraction ability with PointNet++ [2], which is designed to capture different scaled point features. PCAN [5] further improve the place recognition accuracy by modifying the VLAD layer with additional attention module, which can evaluate the point weighting in the feature aggregation.

## III. Proposed Method

As shown in Fig. 3, our method include two modules: a multi-view generation to obtain different perspectives, and the deep fusion network to deeply fuse the multi-view features into a global place descriptor. The single LiDAR scan may effect the place recognition accuracy, our multi-view generation module can generate dense and stable top-down and spherical projections by accumulate LiDAR scans and taking account of updating noises. Different projections share different properties. Fig. 2 shows the geometry changes of the dense local map, top-down view and spherical-view, under locally translation or orientation differences. As we can see that, under the local translation difference, the geometry structure of top-down views are relative stable than spherical-views; with orientation difference, the geometry changes in spherical-view will not change the local feature extraction via convolution networks, while it will affect the feature extraction in the top-down view branch. Our fusion operation can enhance the learnt feature capturing both translation-insensitive propriety of top-down view features and rotation-invariant propriety of spherical-view features.

### A. Multi-View Generation

We utilize the octree structure to cache the local map. Following the original Octomap [16], we apply a log-odds function $L(n)$ to describe a probability distribution $P(n), n \in \mathcal{R}^3$, where we have $L(n) = \log[\frac{P(n)}{1-P(n)}]$. As stated in [16], if the initial occupancy $P(n)$ is set to 0.5, the log-odds $L(n|z_{1:t})$ can be estimated by,

$$L(n|z_{1:t}) = L(n|z_{1:t-1}) + L(n|z_t) \tag{1}$$

In the measurement updating step, observation uncertainty may introduce additional occupancy noise. We cope with such noise by modifying the log-odds updating function. For each node updating factor $L(n|z_t)$, we define a weighting function $\mathcal{W}(z_t|p_t, q_{p_t})$ to measure the points' dynamic property,

$$\mathcal{W}(z_t|p_t, p_{t-1}) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\|p_t - q_{p_t}\|^2/2\sigma^2} \tag{2}$$

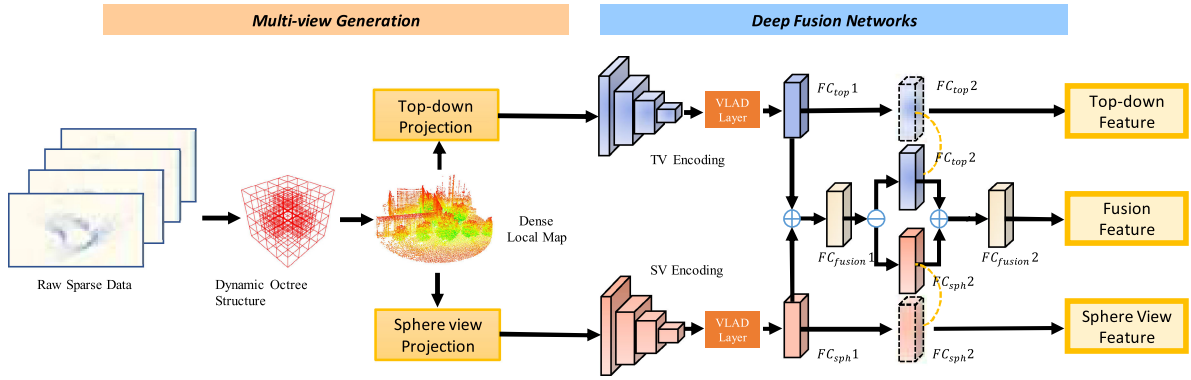


Fig. 3. **The network structure of our method.** Our FusionVLAD mainly includes two modules: Multi-view Generation module and Deep Fusion Networks module. With the given sequence of LiDAR scans, the first module can generate multi-view representations with the consideration of observation and odometry errors. The Deep Fusion Networks can parallel encode the top-down view and spherical-views, and tightly coupled the output features.

where $p_t$ is point within current LiDAR scan $S_T$, and $q_{p_t}$ is the nearest point of $p_t$ in the last LiDAR scan $S_{T-1}$, $\sigma$ is a constant parameter. The original log-odds updating function Eq. (1) is transformed into,

$$L(n|z_{1:t}) = L(n|z_{1:t-1}) + L(n|z_t) \cdot \mathcal{W}(z_t|p_t, q_{p_t}) \quad (3)$$

In the motion updating step, the occupancy of each leaf node (from map $M_{T-1}$ to map $M_T$) is updated with a motion error model $\mathcal{N}(0, \sigma_T)$. The covariance $\sigma_{t_p}$ can be evaluated based on the LiDAR odometry [13] estimation. Thus, we can update the log-odds during the motion updating step by

$$L(n|z_{1:t}) = L(n|z_{1:t}) \cdot \mathcal{N}_M(0, \sigma_{t_p}) \quad (4)$$

The above mechanisms enhance the robot setting a higher belief when the observation is relative accurate and odometry is more stable, or lower belief on the contrary. Then, as shown in Fig. 2 we can generate multi-view representation by projecting the 3D point-cloud from the octree map into the top-down view and the spherical-view respectively.

*1) Top-Down View Projection:* The top-down view projection is encoded with height and intensity values, where the intensity can reflect the materials' properties. To avoid the *pitch* and *roll* difference in top-down projections, we first estimate the major 2D plane on the $z$-axis via RANSAC [17], and project the 3D points onto the major plane with $1\tilde{}m$ resolution level. For each grid on the 2D plane, we use the maximum height of the points within this grid as the height feature, and average intensity as the intensity feature.

*2) Spherical View Projection:* For the spherical projections, we encode the geometry information on the spherical-views. However, different from single LiDAR scan, the point cloud we used are cached into a dense 3D map via the octree data structure. Given a 3D point $p = (x, y, z)$, the relative Zenith and Azimuth angle on the spherical-view can be estimated by,

$$R_Z = \arctan(y, x) \quad (5)$$

$$R_A = \arctan(z, \sqrt{x^2 + y^2}) \quad (6)$$

Within the same range of spherical projection, we may encounter points with different distances. To capture more abundant geometric structures from the 3D environment, the point cloud is divided equally into $M$ slices based on the points' ranges. In our current configuration, the maximum point's distance is 30 meters and $M = 5$. Thus the points within range $0 \sim 6$ are projected on $m_1$, and points within range $24 \sim 30$ are projected on $m_5$. Similar to [18], given a desired resolution $K$, we generate the 3D grids with size of $[K \times K \times M]$. On the grid $(R_{Zi}, R_{Aj})$, we set $d(R_{Zi}, R_{Aj})$ as distance to nearest points within this grid. We also compute the angle $\alpha(R_{Zi}, R_{Aj})$ between the ray and the surface normal at the intersecting face. Thus, for each grid, we get two channels to extract the geometric features on the spherical view.

### B. Deep Fusion Networks

The overall of our FusionVLAD network is shown in Fig. 3. We use two separate 2D convolution networks to extract the local features. Both top-down view ($[K \times K \times 2]$) and spherical-view ($[K \times K \times M \times 2]$) are fed into the convolution layers. In our application, we set $K = 64$ and $M = 5$. In both the spherical-view encoding (SVE) and the top-down view encoding (TVE) branch, the convolution operation for each branch are following the convolution layers in VGG16. By utilizing the VLAD layer [15], we extract the 512-dimension place features from top-down view ($f_{top}$) and Spherical view ($f_{sph}$) separately. To combine information from different representations, we design a tightly coupled fusion network between the top-down and spherical views. As we can see in Fig. 3, with the extracted place features from different perspectives, we obtain the joint feature $f^0_{joint}$ by applying the fully connected layer $FC_1$ on the concatenate features $[f_{top}, f_{sph}]$. Based on the joint feature $f^0_{joint}$, we utilize two fully connected layer $FC^2_{top}$ and $FC^2_{sph}$ to estimate the feature mapping function $F_{top}(\hat{f}_{top}|f_{top}, f_{sph})$ and $F_{sph}(\hat{f}_{sph}|f_{top}, f_{sph})$. To keep the recognition property in top-down and spherical feature respectively, we use the same shared weights fully connected networks $FC^2_{top}$ and $FC^2_{sph}$ in the top-down and spherical branches respectively. For the fully connected networks $FC^1_{top}$, $FC^2_{top}$, $FC^1_{sph}$ and $FC^1_{sph}$, both input and output filter size is 512. For the fully connected networks $FC^1_{fusion}$, $FC^2_{fusion}$, they have the same input filter at 1024, but different output filter sizes (1024 and 512 respectively).

Learning place descriptor differences from the joint feature distributions $[f_{top}, f_{sph}]$ is more complicated than from isolated perspectives. In the training procedure, our share-weights fully

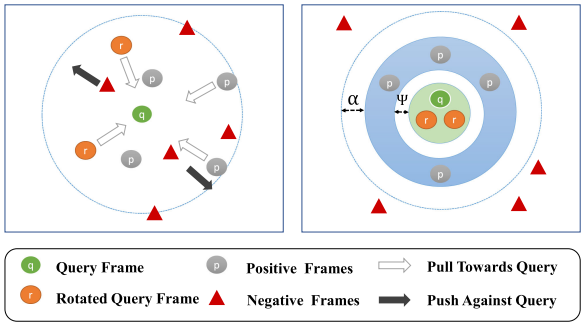


Fig. 4. **Lazy Viewpoint-free metric enhance place feature learning.** The proposed "lazy viewpoint" loss can enhance the margin $\psi$ between the query frame and rotated-query frames, and the margin $\alpha$ between the positive frames (with small translation difference) and negative frames (with large translation difference).

connected layers can build the inter-connections between the isolated top-down/spherical feature distribution and joint feature distribution. In the next section, we will introduce the metric loss that helps the fusion branch learn viewpoint-invariant place descriptors.

### C. Lazy Viewpoint Learning Metric

We propose a novel supervision objective *Lazy viewpoint-free loss* metric to learn the viewpoint-invariant and distinguishable place descriptors. For convenience of illustrating the loss functions, some necessary definitions are first described in following. As we can see in Fig. 4, each training tuple consists of four components: $\mathcal{S} = [S_q, \{S_{rot}\}, \{S_{pos}\}, \{S_{neg}\}]$, where $S_q$ is the query frame of a local 3D scan, $\{S_{rot}\}$ is the rotated query set obtained by rotating $S_q$ with random heading angles ($[0°, 30°, \ldots 330°]$). Similar to the PointNetVLAD [3], $\{S_{pos}\}$ denotes a set of local map representations of 3D scans ("positive") whose mapping center to $\{S_q\}$ is within 5 m; and $\{S_{neg}\}$ denotes a set of 3D scans ("negative") whose mapping center to $\{S_q\}$ is beyond 10 m. To reduce feature difference between $S_q$ and $S_{rot}$, we define the following "lazy rotation" triplet loss,

$$\begin{aligned} &L_{LazyRot}(\mathcal{S}) \\ &= \max_{i,j}([\psi + d(f(S_q), f(S^i_{rot})) - d(f(S_q), f(S^j_{pos})) \\ &= \max_{i,j}([\psi + \delta^i_{rot} - \delta^j_{pos}]_+) \end{aligned} \tag{7}$$

here $f(.)$ is the function that encodes LiDAR representations into the global descriptors. $d(\cdot)$ denoted the squared Euclidean distance between features, and $[.]_+$ denotes the hinge loss. $\psi$ is a constant parameter to mark corresponding margin between $S_{rot}$ and $S_{pos}$. The above triplet loss $L_{LazyRot}$ can also combine with $d(f(S^i_{rot}), f(S^j_{pos}))$ and $d(f(S^i_{rot}), f(S^k_{neg}))$ to further extend the orientation-invariant property. Similarly to NetVLAD [15], we also apply a transitional triplet loss to reduce feature difference between $f(S_q)$ and $f(S_{pos})$ by,

$$\begin{aligned} &L_{LazyTrans}(\mathcal{S}) \\ &= \max_{i,j}([\alpha + d(f(S_q), f(S^i_{pos})) - d(f(S_q), f(S^j_{neg})) \\ &= \max_{i,j}([\alpha + \delta^i_{pos} - \delta^j_{neg}]_+) \end{aligned} \tag{8}$$

where $\alpha$ is constant threshold to control the relative margin. Thus the "lazy viewpoint" loss function $L_{LazyView}$ can be written as,

$$L_{LazyView} = L_{LazyTrans} + L_{LazyRot} \tag{9}$$

In the training procedure, we can train the FusionVLAD in the deep version: applying the "lazy viewpoint" loss either all branches (top-down view-, spherical view-, and fusion-branches); or non-deep version: only applying the "lazy viewpoint" loss on the Fusion branch. In the experiment part, we will investigate the performance of different FusionVLAD configurations. We use the Adam as the optimizer with the initial learning rate at $1e-4$.

## IV. Experiments

In this section, we compare the proposed method with the current state-of-the-art in learning-based 3D place recognition on both public and self-recorded datasets. To generate our datasets, we designed a data recording mobile platform. The platform contains a LiDAR device (Velodyne-VLP 16), an inertial measurement unit (Xsense MTi 30, $0.5°$ error in roll/pitch, $1°$ error in yaw, $550\tilde{}mW$), a mini PC (Intel NUC i7, 3.5 GHz, 28 W) and an embedded GPU device (Nvidia Xavier, 8 G memory). We trained our network model on a GPU server with a Ubuntu 18.04 system with a single Nvidia 1080Ti GPU and 64 G RAM. In the rest of this section, we will first describe both the public and self-recorded datasets. Then we analyze the place recognition performance and efficiency of our methods, other state-of-the-art learning-based methods [3]–[5], and non-learning based methods [10], [12], [19], [20]. Finally, in the ablation study section, we also analyze our methods' performance under different network configurations.

**Datasets:** Our evaluation datasets include two public datasets and two self-recorded datasets:

- **KITTI** odometry datasets, which consists of 21 trajectories generated with Velodyne-64 LiDAR scanner, around the mid-size city of Karlsruhe. We use trajectory $\{1 \sim 8\}$ for network training, and $\{9, 10\}$ for evaluation.
- **NCLT** datasets, which consists of 103 trajectories generated with Velodyne-32 LiDAR scanner, around the mid-size city of Karlsruhe. We use trajectory $\{1 \sim 8\}$ for network training, and $\{9, 10\}$ for evaluation.
- **Campus dataset**, we created a *Campus* dataset with 11 trajectories by driving our data generating mobile platform to traverse a 2 km outdoor route around the campus of Carnegie Mellon University.
- **City dataset**, we created a City-scale dataset by mounting our data generating system on the top of a car and repeatly traversing $11\tilde{}km$ trajectories in the city of Pittsburgh. We collected 12 trajectories, and we use trajectories $\{1 \sim 10\}$ for network training, and $\{11, 12\}$ for evaluation.

To provide training/evaluation data, we first generate the global map with all the LiDAR sequences through a traditional LiDAR odometry method [13]. We generate the local LiDAR maps given the ground truth trajectories on each datasets, and separate them into training and evaluation datasets with shuffled index. The train/inference datasets are gathered individually, and we only train the network with the specific training dataset in each dataset. We also prepare the corresponding position for

TABLE I
DATASET FRAMES NUMBER IN TRAINING/EVALUATION ON THE FOUR DATASETS

| | *KITTI* | *NCLT* | *CAMPUS* | *CITY* |
|---|---|---|---|---|
| Train | 13,070 | 15,971 | 13,830 | 10,972 |
| Eval (test/refer) | 3,268 | 3,993 | 3,458 | 2,744 |

TABLE II
PLACE RECOGNITION EVALUATION ON DIFFERENT AREAS OF *CMU*

| | *Refer* | *Test* | *Distance* |
|---|---|---|---|
| *NSH* | 1,523 | 1,489 | $1.5km$ |
| *Hamburg Hall* | 1,830 | 1,865 | $1.8km$ |
| *Gates* | 2,023 | 1,948 | $2.0km$ |

each local map, which is necessary to calculate the position $\{S_{pos}\}$ and negative $\{S_{neg}\}$ samples for each query data $S_q$. Table I shows the data separation in training/evaluation on the four datasets. Especially, to verify the generalization ability of different methods in the evaluation step, we generate the testing data with random delta translation ($[0, 2, 4, 6, 8, \ldots]m$ on 2D–XY plane) or delta orientation ($[0, 45, 90\,135, \ldots]^\circ$ in yaw) differences comparing the ground truth trajectory.

On the other hand, to verify the place recognition performance of two slightly different trajectories within the same area, we also record the testing/reference sequence data from different areas on the campus area as we can see in the Table II. The ground truth positions of testing/reference sequence queries are obtained with an offline ICP-based point cloud registration method, where each frame will take around 200 $\sim$ 500 ms with accuracy around 0.1 $\sim$ 0.2 m.

**Methods:** To evaluate our approach's performance, we firstly compared it with the current state-of-the-art methods of learning-based place recognition methods, PointNetVLAD [3], which are re-implied from the Github.[1] For the non-learning based method, we also compare the frame-level descriptors Fast Histogram [20] and M2DP [12] from the Github,[2] and 3D keypoint-based methods ESF [19] and SHOT [10] in the PCL library. For all the above approaches, we provide the dense local maps generated by our local mapping module, as shown in Section III-A. For the all the learning-based method, we use the same data configuration on each datasets and same training epochs, and training the new network models on each of four datasets. To inference different real trajectories in the Campus (Table II), we use the pre-trained model on the joint Campus dataset.

**Evaluation Manners and Metrics:** We evaluate the place recognition accuracy under two different matching manners: single-scan and sequence-scan based place recognition. The first manner retrieves corresponding places by evaluating feature similarity with K-nearest neighbor search. The second manner follows sequence-matching based place recognition in SeqSLAM [21]. Here we only use the sequence matching steps in the original SeqSLAM. For both manners, similar to PointNetVLAD [3], we use the Average Recall, Precision-Recall Curve and Average Precision Score (AP score) to assess the place recognition accuracy between the relative reference and testing sequence trajectories.

[1]https://github.com/mikacuy/pointnetvlad
[2]https://github.com/LiHeUA/M2DP

Fig. 5. **Sub-map generation.** Each sub-map (the highlighted area) is generated from the global map $Map_i$. In the evaluation step, both testing and reference queries are extracted with local viewpoints differences in translation ($[0, 2, 4, 6, \ldots]m$ on 2D–XY plane) and orientation ($[0, 45, 90\,135, \ldots]^\circ$) in yaw.

TABLE III
AVERAGE RECALL AT TOP @5 ON FOUR DATASETS UNDER VARIANT VIEWPOINT DIFFERENCES FOR BOTH NON-LEARNING/LEARNING METHODS

| Method | *KITTI* | *NCLT* | *Campus* | *CITY* |
|---|---|---|---|---|
| Non-Learning Methods | | | | |
| M2DP | 83.83% | 27.88% | 72.50% | 60.59% |
| ESF | 81.84% | 82.38% | 84.66% | 82.67% |
| Fast-histogram | 35.07% | 17.05% | 24.09% | 24.69% |
| SHOT | 42.84% | 41.77% | 53.86% | 58.89% |
| Learning Methods | | | | |
| PointNetVLAD | 32.13% | 33.56% | 35.48% | 32.35% |
| Fusion (top-down) | 78.88% | 82.15% | 78.05% | 75.54% |
| Fusion (spherical) | 74.49% | 79.24% | 73.22% | 71.37% |
| Fusion (no deep) | 82.31% | 85.23% | 81.76% | 79.21% |
| Fusion (deep) | **86.07%** | **94.53%** | **89.61%** | **86.64%** |
| Fusion (without rot) | 80.29% | 83.92% | 80.41% | 76.31% |

## A. Place Recognition Results

*1) Orientation- and Translation- Tolerance Analysis:* We conduct experiments on four datasets to evaluate the robustness of place recognition of different methods. As we can see in Fig. 6, we calculate the average recall at top 1% between reference/testing trajectories (under translation $[0, 2, 4, 6]m$ and yaw orientation $[0, 45, 90\,135]^\circ$). For all the non-learning based approaches, they can achieve robustly place recognition under variant orientation difference, while on the other hand, translation difference will significantly reduce their matching accuracy. We can also notice that the PointNet-based method can handle translation differences locally but is sensitive to orientation differences, the matching accuracy of PointNetVLAD quickly declined as rotation differences increasing. Compare to the above techniques, the place descriptors of FusionVLAD are invariant to orientation-differences and can guarantee the average recall at top 1% are above 80% on each dataset.

Table III further provides the average recall at top @5, we can see that on all the four datasets. Here Fusion (top-down) and Fusion (spherical) indicates the network with only one perspective. As we mentioned in Section III-C, Fusion (no deep) means FusionVLAD network but only with fusion-branch loss in the training procedure, and Fusion (deep) is with all the three branches. Fusion (without rot) means the training procedure without 'rotation loss' metric. FusionVLAD can achieve more accurate place retrieval with average recall above 85%. Table IV

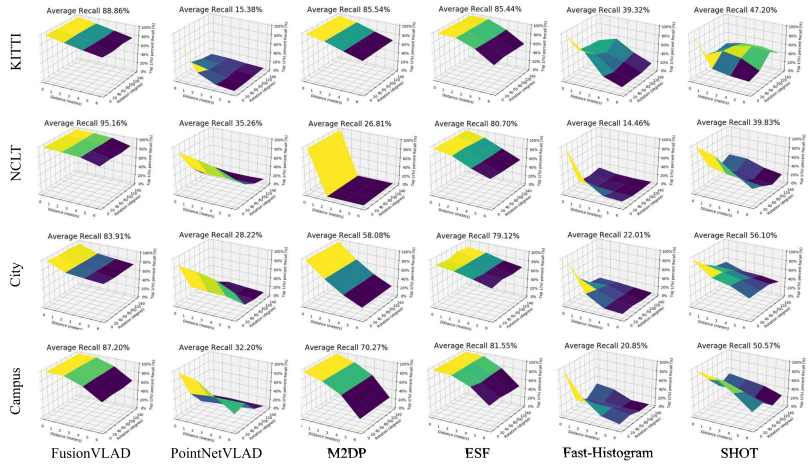


Fig. 6. **Average Recall at Top 1% under variant viewpoint differences.** Each sub-graph represent 16 trails of the place recognition average recall (z-axis) on different translation (x-axis) or orientations (y-axis). Each trail includes a reference trajectory and a testing trajectory (with translation/orientation noise).

TABLE IV
AVERAGE RECALL AT TOP @5 UNDER DIFFERENT AREAS OF THE CAMPUS

| Method | *NSH* | *Hamburg Hall* | *Gates* |
|---|---|---|---|
| Non-Learning Methods | | | |
| M2DP | 65.38% | 75.29% | 72.15% |
| ESF | 80.14% | 89.23% | 86.27% |
| Fast-histogram | 32.15% | 36.94% | 35.74% |
| SHOT | 48.41% | 51.58% | 45.36% |
| Learning Methods | | | |
| PointNetVLAD | 67.59% | 72.71% | 69.16% |
| Fusion (top-down) | 80.81% | 79.59% | 81.36% |
| Fusion (spherical) | 75.27% | 80.24% | 76.82% |
| Fusion (no deep) | 83.86% | 85.12% | 86.32% |
| Fusion (deep) | **89.51%** | **92.16%** | **88.14%** |
| Fusion (without rot) | 82.73% | 81.08% | 83.65% |

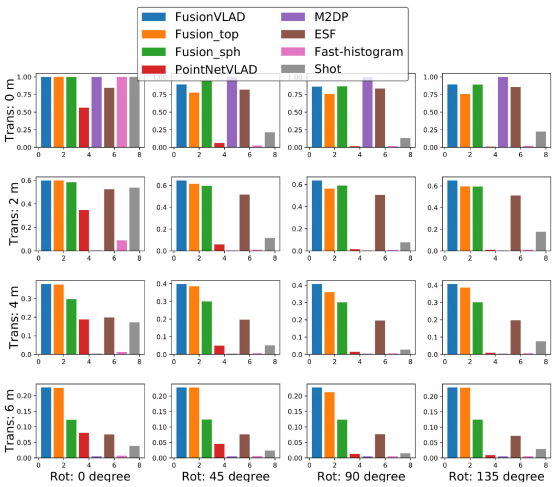


Fig. 7. Average precision score of single-matching results on the NCLT dataset.

shows the place recognition performance on the datasets of II, and all the learning methods use the pre-trained model on the Campus datasets.

*2) Single- and Sequence-Matching Results:* Fig. 7 and Fig. 8 shows the place recognition results of single-scan and

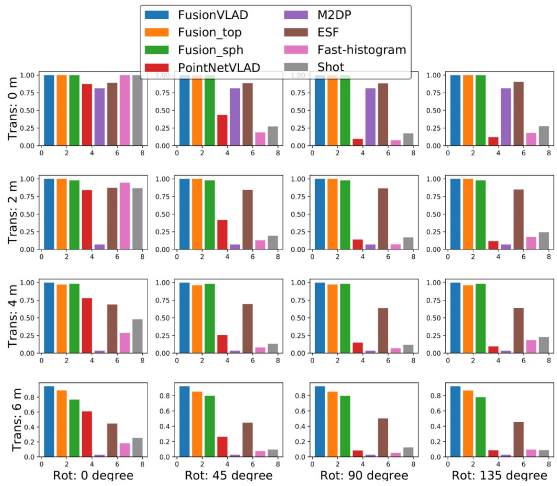


Fig. 8. Average precision score of sequence-matching [21] results on the NCLT dataset.

sequence-scan based matching on the *NCLT* dataset. In both matching manners, we use the precision-recall curve and Average Precision Score to analyze the matching accuracy. In the cases of lower translation differences, both non-learning and learning-based methods can achieve robust place recognition ability. For significant translational or rotational differences, transitional methods or PointNet-based method cannot guarantee an accurate place retrieval. In most cases, the sequence-matching manner can improve the matching performances, however, for M2DP method without translation differences, sequence-matching may reduce the recognition accuracy. For FusionVLAD, with the benefit of merging both top-down view and spherical-view branches, our method can achieve stable place retrieval in both single- and sequence- matching manner. Especially, we can achieve 80% Average Precision Score under sequence-matching manner within $6\tilde{}m$ translation differences. Please not the sequence-matching can only find one single best matching for each query, thus this manner need to further modified when dealing with query with more than two matches.

TABLE V
COMPARISON OF TIME, GPU MEMORY (MEGABYTE), AND FEATURE SIZE REQUIREMENTS OF DIFFERENT METHODS

| Method | GPU (MB) | Time (ms) | Feature Size |
|---|---|---|---|
| Non-Learning Methods | | | |
| *M2DP* | – | 164.05 | 192 |
| *ESF* | – | 14.56 | 640 |
| *Fast-histogram* | – | 1.47 | 80 |
| *SHOT* | – | 14.74 | 352 |
| Learning Methods | | | |
| *PointNetVLAD* | 7,711 | 55.00 | 256 |
| *FusionVLAD* | 2,459 | 10.00 | 256 |

### B. Time and Storage Analysis

In Table V, we analysis the time and memory usage of our FusionVLAD and other methods. Compared with PointNetVLAD, our method consumes about 31% GPU memory for training and takes about 20% time for extracting place descriptor for a point cloud. While comparing to non-learning method, FusionVLAD is more robust and efficient. In the inference step, the multi-view generation step will consume $100\tilde{}ms$ on the NUC i7 computer. FusionVLAD also only requires $10\tilde{}ms$ and $37Mb$ on the GPU device or $87\tilde{}ms$ on a CPU device. which make it easy to imply in the Nvidia embedded system (Nvidia Jetson TX2, or Xavier) or the NUC computer, and suitable for low cost robot systems in long-term SLAM or navigation tasks.

### C. Discussion & Limitations

As shown in the above analysis, the performance of FusionVLAD can benefit from multi-view perspectives. However, in the situation of confined spaces (e.g., indoors, tunnels and underground), the generated LiDAR map is very sparse and cannot generated rich textures from top-down view or spherical views. One potential solution for the confined spaces is using camera to increase the depth resolution, which may has a computation-expensive operation in depth prediction. Another limitation comes from the RANSAC-based plane fitting for top-down view generation, which is only suitable for structured ground.

We also evaluate the place recognition performance with single LiDAR scans instead of the cached maps from octree. The performance of both learning/non-learning methods are depended on the resolution of LiDAR device, and all methods perform bad with single VLP-16 scans; thus our multi-view generation module is necessary in this case.

## V. CONCLUSION

In this letter, we designed a parallel fusion network structure, FusionVLAD, to learn robust 3D place descriptors from multi-view LiDAR projections. Our method can simultaneously extract features from the top-down and the spherical-view branch, and we design a deep fusion layer to guarantee the extracted descriptors inherit both translation-insensitive and orientation-invariant properties. The results on both public and self-recorded datasets show that our method notably outperforms the state-of-the-art in 3D place recognition tasks. Our method can work with limited computation resources and storage spaces, making it extremely suitable for low-cost robots in large-scale mapping tasks. In future works, we will fuse information from different source (image, radar) to further improve the performance.

## REFERENCES


[1] 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. Pattern Recognit.*, 2017, pp. 652–660.

[2] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet : Deep hierarchical feature learning on point sets in a metric space," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5099–5108.

[3] 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.*, Jun 2018, pp. 4470–4479.

[4] Z. Liu *et al.*, "Lpd-net: 3d point cloud learning for large-scale place recognition and environment analysis," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 2831–2840.

[5] W. Zhang and C. Xiao, "Pcan: 3d attention map learning using contextual information for point cloud based retrieval," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 12 436–12 445.

[6] D. G. Lowe, "Object Recognition From Local Scale-Invariant Features," in *Proc. 7th IEEE Int. Conf. Comput. Vis.*, vol. 2, 1999, pp. 1150–1157.

[7] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Eur. Conf. Comput. Vis.* Berlin, Germany: Springer, 2006, pp. 404–417.

[8] W. Wohlkinger and M. Vincze, "Ensemble of shape functions for 3 d object classification," in *Proc. IEEE Int. Conf. Robot. Biomimetics*., 2011, pp. 2987–2992.

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

[10] F. Tombari, S. Salti, and L. Di Stefano, "Unique signatures of histograms for local surface description," in *Proc. Eur. Conf. Comput. Vis.* Berlin, Germany: Springer, 2010, pp. 356–369.

[11] T. Röhling, J. Mack, and D. Schulz, "A fast histogram-based similarity measure for detecting loop closures in 3-d lidar data," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 736–741.

[12] 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.

[13] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time," *Robot.: Sci. Syst.*, vol. 2, p. 9, 2014.

[14] R. Dubé, D. Dugas, E. Stumm, J. Nieto, R. Siegwart, and C. Cadena, "Segmatch: Segment based place recognition in 3 d point clouds," in *Proc. IEEE Int. Conf. Robot. Automat.*, May 2017, pp. 5266–5272.

[15] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "Netvlad: Cnn architecture for weakly supervised place recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 5297–5307.

[16] E. Rho and S. Jo, "Octomap-based semi-autonomous quadcopter navigation with biosignal classification," in *Proc. 6th Int. Conf. Brain-Comput. Interface*, Jan. 2018, pp. 1–4.

[17] R. Schnabel, R. Wahl, and R. Klein, "Efficient ransac for point-cloud shape detection," *Computer Graphics Forum*, vol. 26, no. 2. Wiley Online Library, 2007, pp. 214–226.

[18] C. Esteves, C. Allen-Blanchette, A. Makadia, and K. Daniilidis, "Learning so (3) equivariant representations with spherical cnns," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 52–68.

[19] W. Wohlkinger and M. Vincze, "Ensemble of shape functions for 3d object classification," in *Proc. IEEE Int. Conf. Robot. Biomimetics*, 2011, pp. 2987–2992.

[20] T. Röhling, J. Mack, and D. Schulz, "A fast histogram-based similarity measure for detecting loop closures in 3-d lidar data," in *2015 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 736–741.

[21] M. J. Milford and G. F. Wyeth, "SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights," in *Proc. IEEE Int. Conf. Robot. Automat.*, May 2012, pp. 1643–1649.