

DH3D: Deep Hierarchical 3D Descriptors for Robust Large-Scale 6DoF Relocalization

Juan Du^{1*}, Rui Wang^{1,2*}, and Daniel Cremers^{1,2}

¹ Technical University of Munich

² Artisense

{duj, wangr, cremers}@in.tum.de

Abstract. For relocalization in large-scale point clouds, we propose the first approach that unifies global place recognition and local 6DoF pose refinement. To this end, we design a Siamese network that jointly learns 3D local feature detection and description directly from raw 3D points. It integrates FlexConv and Squeeze-and-Excitation (SE) to assure that the learned local descriptor captures multi-level geometric information and channel-wise relations. For detecting 3D keypoints we predict the discriminativeness of the local descriptors in an unsupervised manner. We generate the global descriptor by directly aggregating the learned local descriptors with an effective attention mechanism. In this way, local and global 3D descriptors are inferred in one single forward pass. Experiments on various benchmarks demonstrate that our method achieves competitive results for both global point cloud retrieval and local point cloud registration in comparison to state-of-the-art approaches. To validate the generalizability and robustness of our 3D keypoints, we demonstrate that our method also performs favorably without fine-tuning on the registration of point clouds that were generated by a visual SLAM system. Code and related materials are available at <https://vision.in.tum.de/research/vslam/dh3d>.

Keywords: Point clouds · 3D deep learning · Relocalization

1 Introduction

Relocalization within an existing 3D map is a critical functionality for numerous applications in robotics [3] and autonomous driving [35, 53]. A common strategy is to split the problem into two subtasks, namely global place recognition and local 6DoF pose refinement. A lot of effort has been focused on tackling the problem using 2D images [6, 44, 46, 51], where the 3D maps are usually defined as image feature points reconstructed in 3D using Structure from Motion (SfM). The coarse global place recognition is achieved by image retrieval, whereas accurate local 6DoF pose refinement is addressed separately by feature matching and PnP. With the progress of deep learning for image descriptor extraction [2, 19]

* Authors contributed equally.

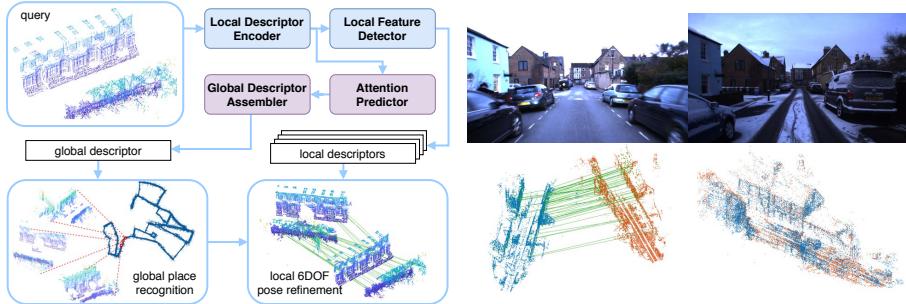


Fig. 1: Left: We propose a hierarchical network for large-scale point cloud based relocalization. The network consumes raw 3D points and performs local feature detection, description and global descriptor extraction in one forward pass. The global descriptor is used to retrieve similar scenes from the database. Accurate 6DoF pose is then obtained by matching the local features. Right: Our local descriptors trained on LiDAR points work favorably on the sparse point clouds generated by a visual SLAM method without fine-tuning. The point clouds are generated from sequences with different weathers, lighting conditions and scene layouts, thus *have significantly different distributions*.

and 2D keypoints detection/description [14, 15, 40, 60], image based methods have significantly gained in robustness to variations in viewpoint and illumination.

Alternatively one can tackle these variations by working on 3D point clouds since these are inherently invariant to such issues. Moreover, there exist numerous SLAM pipelines that generate accurate large-scale point clouds using sensory input from LiDAR [13, 63, 64] or camera [17, 55]. While there is great potential to rely on such data, research on point cloud based relocalization is significantly less matured compared to the image-based counterpart [15, 40]. Especially, deep learning on 3D descriptors emerged only roughly 3 years ago. With most of the early attempts focusing on small-scale tasks like object classification, detection and segmentation [28, 37, 39, 57, 68], only a limited number of networks have been proposed for large-scale localization [30, 31, 58]. Moreover, among these few attempts, global place recognition [1, 65] and local 6DoF pose refinement [8, 18, 23] have been addressed isolatedly, despite the fact that both tasks depend on the same low level geometric clues.

In this paper, we propose a hierarchical deep network for large-scale point clouds based relocalization – see Fig. 1. The network directly consumes unordered 3D points and performs keypoint detection and description, as well as global point cloud descriptor extraction in a unified manner. In contrast to the conventional *detect-then-describe* pipeline, our local features are learned with the *detect-and-describe* concept. We estimate a confidence map of the discriminativeness of local features explicitly and learn to select keypoints that are well-suited for matching in an unsupervised manner. The local features are aggregated into a global descriptor for global retrieval, attaining a consistent workflow for large-scale outdoor 6DoF relocalization. Our main contributions are summarized as follows:

- We propose the first work that unifies point cloud based global place recognition and 6DoF pose refinement. Our method performs local feature detection and description, as well as global descriptor extraction in one forward pass, running significantly faster than previous methods.
- We propose to use FlexConv and SE block to integrate multi-level context information and channel-wise relations into the local features, thus achieve much stronger performance on feature matching and also boost the global descriptor.
- We introduce a *describe-and-detect* approach to explicitly learn a 3D keypoint detector in an unsupervised manner.
- Both our local and global descriptors achieve state-of-the-art performances on point cloud registration and retrieval across multiple benchmarks.
- Furthermore, our local descriptors trained on LiDAR data show competitive generalization capability when applied to the point clouds generated by a visual SLAM method, even though LiDAR and visual SLAM point clouds exhibit very different patterns and distributions.

2 Related Work

Handcrafted local descriptors encode local structural information as histograms over geometric properties e.g., surface normals and curvatures. Spin image (SI) [24] projects 3D points within a cylinder onto a 2D spin image. Unique Shape Context (USC) [52] deploys a unique local reference frame to improve the accuracy of the well-known 3D shape context descriptor. Point Feature Histogram (PFH) [43] and Fast PFH (FPFH) [42] describe the relationships between a point and its neighbors by calculating the angular features and normals. While these handcrafted methods have made great progress, they generalize poorly to large-scale scenarios and struggle to handle noisy real-world data.

Learned local descriptors. To cope with the inherent irregularity of point cloud data, researchers have suggested to convert 3D points to regular representations such as voxels and multi-view 3D images [38, 47, 49, 50, 54]. As a pioneering work, PointNet [37] proposed to apply deep networks directly to raw 3D points. Since then, new models [28, 39, 66] and flexible operators on irregular data [21, 56, 59] have been emerging. Accompanied by these progresses, learning-based 3D local descriptors such as 3DMatch [62], PPFNet [11] and PPF-FoldNet [10, 12] have been proposed for segment matching, yet they are designed for RGB-D based indoor applications. Recently, Fully Convolutional Geometric Features (FCGF) [8] was proposed to extract geometric features from 3D points. Yet, all these methods do not tackle feature detection and get their features rather by sampling. Another class of methods utilizes deep learning to reduce the dimensions of the handcrafted descriptors, such as Compact Geometric Features (CGF) [25] and LORAX [16]. In the realm of large-scale outdoor relocalization, 3DFeatNet [23] and L³-Net [31] extract local feature embedding using PointNet, whereas 3DSmoothNet [18] and DeepVCP [30] rely on 3D CNNs. In contrary to registration based on feature matching, DeepVCP and Deep Closest Point [58] learn to locate the correspondences in the target point clouds.

3D keypoint detectors. There are three representative hand-crafted 3D detectors. Intrinsic Shape Signatures (ISS) [67] selects salient points with large variations along the principal axes. SIFT-3D [29] constructs a scale-space of the curvature with the DoG operator. Harris-3D [48] calculates the Harris response of each 3D vertex based on first order derivatives along two orthogonal directions on the 3D surface. Despite the increasing number of learning-based 3D descriptors, only a few methods have been proposed to learn to detect 3D keypoints. 3DFeatNet [23] and DeepVCP [30] use an attention layer to learn to weigh the local descriptors in their loss functions. Recently, USIP [27] has been proposed to specifically detect keypoints with high repeatability and accurate localization. It establishes the current state of the art for 3D keypoint detectors.

Handcrafted global descriptors. Most 3D global descriptors describe places with handcrafted statistical information. Rohling et al. [41] propose to describe places by histograms of points elevation. Cop et al. [9] leverage LiDAR intensities and present DELIGHT. Cao et al. [5] transform a point cloud to a bearing-angle image and extract ORB features for bag-of-words aggregation.

Learned global descriptors. Granström et al. [20] describe point clouds with rotation invariant features and input them to a learning-based classifier for matching. LocNet [61] inputs range histogram features to 2D CNNs to learn a descriptor. In these methods, deep learning essentially plays the role of post-processing the handcrafted descriptors. Kim et al. [26] transform point clouds into scan context images and feed them into CNNs for place recognition. PointNetVLAD [1] first tackles place recognition in an end-to-end way. The global descriptor is computed by a NetVLAD [2] layer on top of the feature map extracted using PointNet [37]. Following this, PCAN [65] learns attentions for points to produce more discriminative descriptors. These two methods extract local features using PointNet, which projects each point independently into a higher dimension and thus does not explicitly use contextual information.

3 Hierarchical 3D Descriptors Learning

For large-scale relocalization, an intuitive approach is to tackle the problem hierarchically in a coarse-to-fine manner: local descriptors are extracted, aggregated into global descriptors for coarse place recognition, and then re-used for accurate 6DoF pose refinement. While being widely adopted in image the domain [33, 44, 45], this idea has not been addressed by the deep learning community for 3D. As a result, seeking for 6DoF relocalization for point clouds, one has to perform local feature detection, description, global descriptor extraction separately, possibly running an independent network for each. To address this problem, we design a hierarchical network operating directly on a point cloud, delivering local descriptors, a keypoint score map and a global descriptor in a single forward pass. Point cloud based relocalization thus can be performed hierarchically: a coarse search using the global descriptor retrieves 3D submap candidates, which are subsequently verified by local 3D feature matching to estimate the 6DoF poses. An overview of our system is provided in Fig. 1.

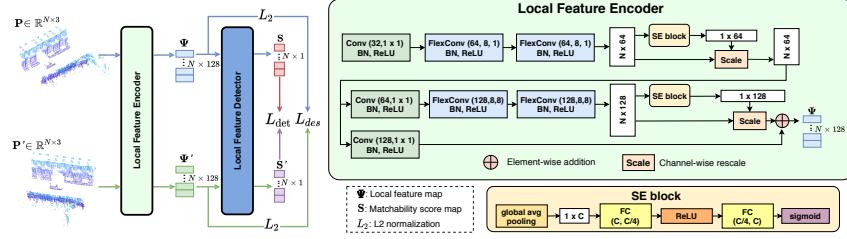


Fig. 2: Left: the flow of local feature descriptor and detector learning. Right: the architecture of our Local Feature Encoder. In $\text{Conv}(D, K \times K)$ and $\text{FlexConv}(D, k, d)$, D : output dimension, K : filter size, k : neighborhood size, d : dilation rate.

3.1 3D Local Feature Encoder and Detector

3DFeatNet [23] is a seminal work that learns both 3D local feature detection and description. Nevertheless, the following two points potentially limit its discriminative power: (1) Its detector is an attention map learned directly from the input points. During inference descriptors are only extracted for the keypoints defined by the attention map. Such classical *detect-then-describe* approach, as discussed in [15, 40], typically focuses on low-level structures of the raw input data, and cannot utilize the high level information encoded in the descriptors. (2) Its feature description is PointNet-based, the symmetric function of which tends to provide only limited structural information of local clusters. To resolve these limitations, we propose to use Flex Convolution (FlexConv) [21] and Squeeze-and-Excitation (SE) block [22] to respectively fuse multi-level spatial contextual information and channel-wise feature correlations into the local descriptors. The *describe-and-detect* pipeline [15, 40] is adopted to postpone our detection stage to employ higher-level information in the learned descriptors.

FlexConv layer. Considering a 3D point \mathbf{p}_l with its k neighbors $N_k(\mathbf{p}_l) = \{\mathbf{p}_{l_1}, \dots, \mathbf{p}_{l_k}\}$ as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{0, \dots, k\}$ are the vertices and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ the edges. A 3D operator on an edge of this graph can be formulated as $\mathbf{e}_{ll_k} = f_{\Theta}(\mathbf{p}_l, \mathbf{p}_{l_k})$, with Θ the set of learnable parameters. As pointed out by [59], PointNet is a special case of $f_{\Theta}(\mathbf{p}_l, \mathbf{p}_{l_k}) = f_{\Theta}(\mathbf{p}_l)$, thus encodes only global shape information and ignores the local neighborhood structure. In contrary, FlexConv can be abstracted as $f_{\Theta}(\mathbf{p}_l, \mathbf{p}_{l_k}) = f_{\Theta}(\mathbf{p}_l - \mathbf{p}_{l_k}, \mathbf{p}_{l_k})$, therefore can effectively encode local information, which we believe is crucial for learning discriminative local descriptors. Formally, FlexConv is a generalization of the conventional grid-based convolution and is defined as:

$$f_{\text{FlexConv}}(\mathbf{p}_l) = \sum_{\mathbf{p}_{l_i} \in N_k(\mathbf{p}_l)} \omega(\mathbf{p}_{l_i}, \mathbf{p}_l) \cdot h(\mathbf{p}_{l_i}), \quad (1)$$

where $h(\mathbf{p}_{l_i}) \in \mathbb{R}^C$ is a point-wise encoding function projecting a point to the high-dimensional feature space. It is convolved with a filter-kernel $\omega : \mathbb{R}^3 \times \mathbb{R}^3 \rightarrow \mathbb{R}^C$ that is computed by the standard scalar product in the Euclidean space, with learnable parameters $\theta \in \mathbb{R}^{C \times 3}$, $\theta_b \in \mathbb{R}^C : w(\mathbf{p}_l, \mathbf{p}_{l_i} | \theta, \theta_b) = \langle \theta, \mathbf{p}_l - \mathbf{p}_{l_i} \rangle + \theta_b$. It can be considered as a linear approximation of the traditional filter-kernel which

uses the location information explicitly. In addition, ω is everywhere well-defined and allows us to perform back-propagation easily.

Squeeze-and-Excitation (SE) block. While FlexConv models spatial connectivity patterns, SE blocks [22] are further used to explicitly model the channel-wise inter-dependencies of the output features from the FlexConv layers. Let $\mathbf{U} = \{u_1, \dots, u_C\} \in \mathbb{R}^{N \times C}$ denote the input feature map to the SE block, where $u_c \in \mathbb{R}^N$ represents the c -th channel vector of the output from the last FlexConv layer. The squeeze operation first “squeezes” \mathbf{U} into a channel-wise descriptor $z \in \mathbb{R}^C$ as $f_{sq} : \mathbb{R}^{N \times C} \rightarrow \mathbb{R}^C$, $z = f_{sq}(\mathbf{U})$, where f_{sq} is implemented as a global average pooling to aggregate across spatial dimensions. $z \in \mathbb{R}^C$ is an embedding containing global information which is then processed by the excitation operation $f_{ex} : \mathbb{R}^C \rightarrow \mathbb{R}^C$, $s = f_{ex}(z)$, where $s \in \mathbb{R}^C$ is implemented as two fully connected layers with ReLU to fully capture channel-wise dependencies and to learn a nonlinear relationship between the channels. In the end, the learned channel activations are used to recalibrate the input across channels achieving the attention selection of different channel-wise features $\tilde{u}_c = f_{scale}(u_c, s_c) = s_c \cdot u_c$, where $\tilde{\mathbf{U}} = \{\tilde{u}_1, \dots, \tilde{u}_C\}$ refers to the output of the SE block.

Encoder architecture. The architecture of the encoder module is illustrated in Fig. 2. In comparison to 3DFeatNet [23] which relies on PointNet and only operates on one level of spatial granularity, our encoder extracts structural information from two spatial resolutions. At each resolution, the following operations are conducted: one 1×1 convolution, two consecutive FlexConv layers and a SE block. Taking a point cloud $\mathbf{P} = \{\mathbf{p}_1, \dots, \mathbf{p}_N\} \in \mathbb{R}^{N \times 3}$ as input, the encoder fuses multi-level contextual information by adding the outputs from the two resolutions and produces the feature map Ψ . It is then L2-normalized to give us the final local descriptor map $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\} \in \mathbb{R}^{N \times D}$. Benefited from the better policies for integrating contextual information, when compared to 3DFeatNet, our local features are much more robust to points densities and distributions, thus generalize significantly better to point clouds generated by different sensors (more details in Sec. 4.4).

Description loss. In feature space, descriptors of positive pairs are expected to be close and those of negative pairs should keep enough separability. Instead of using the simple Triplet Loss as in 3DFeatNet, we adopt the N-tuple loss [11] to learn to differentiate as many patches as possible. Formally, given two point clouds \mathbf{P}, \mathbf{P}' , the two following matrices can be computed: a feature space distance matrix $\mathbf{D} \in \mathbb{R}^{N \times N}$ with $\mathbf{D}(i, j) = \|\mathbf{x}_i - \mathbf{x}_j\|$, a correspondence matrix $\mathbf{M} \in \mathbb{R}^{N \times N}$ with $\mathbf{M}_{i,j} \in \{0, 1\}$ indicating whether two point patches $\mathbf{p}_i \in \mathbf{P}$ and $\mathbf{p}_j \in \mathbf{P}'$ form a positive pair, i.e., if their distance is within a pre-defined threshold. The N-tuple loss is formulated as:

$$L_{desc} = \sum^* \left(\frac{\mathbf{M} \circ \mathbf{D}}{\|\mathbf{M}\|_F^2} + \eta \frac{\max(\mu - (1 - \mathbf{M}) \circ \mathbf{D}, 0)}{N^2 - \|\mathbf{M}\|_F^2} \right), \quad (2)$$

where $\sum^*(\cdot)$ is element-wise sum, \circ element-wise multiplication, $\|\cdot\|_F$ the Frobenius norm, η a hyper-parameter balancing matching and non-matching pairs.

The loss is divided by the number of true/false matches to remove the bias introduced by the larger number of negatives.

3D local feature detection. Contrary to the classical *detect-then-describe* approaches, we postpone the detection to a later stage. To this end, we produce a keypoint saliency map $\mathbf{S} \in \mathbb{R}^{N \times 1}$ from the extracted point-wise descriptors instead of from the raw input points. Our saliency map is thus estimated based on the learned local structure encoding and thus is less fragile to low level artifacts in the raw data, and provides significantly better generalization. Another benefit of the *describe-and-detect* pipeline is that feature description and detection can be performed in one forward pass, unlike the *detect-then-describe* approaches that usually need two stages. Our detector consumes the local feature map Ψ by a series of four 1×1 convolution layers, terminated by the sigmoid activation function (more details in the supplementary document).

As there is no standard definition of a keypoint’s discriminativeness for outdoor point clouds, keypoint detection cannot be addressed by supervised learning. In our case, since the learned descriptors are to be used for point cloud registration, we propose to optimize keypoint confidences by leveraging the quality of descriptor matching. Local descriptor matching essentially boils down to nearest neighbor search in the feature space. Assuming the descriptor is informative enough and the existence of correspondence is guaranteed, a reliable keypoint, i.e., a keypoint with a high score s , is expected to find the correct match with high probability. We therefore can measure the quality of the learned detector using $\eta_i = (1 - s_i) \cdot (1 - \mathbf{M}_{i,j}) + s_i \cdot \mathbf{M}_{i,j}$, where $s_i \in [0, 1]$ is an element of \mathbf{S} and j refers to the nearest neighbor in Ψ' . A simple loss function can be formulated as $L_{det} = \frac{1}{N} \sum_{i=1}^N 1 - \eta_i$. However, we find that only using the nearest neighbor to define η is too strict on the learned feature quality and the training can be unstable. We thus propose a new metric called average successful rate (AR): given a point $\mathbf{p}_i \in \mathbf{P}$ and its feature $\psi_i \in \Psi$, we find the k nearest neighbors in Ψ' . The AR of \mathbf{p}_i is computed as: $AR_i = \frac{1}{k} \sum_{j=1}^k c_{ij}$, where $c_{ij} = 1$ if at least one correct correspondence can be found in the first j candidates, otherwise is 0.³ Now we can measure η with AR which is a real number in the range $[0, 1]$ instead of a binary number and the loss above can be rewritten as:

$$L_{det} = \frac{1}{N} \sum_{i=1}^N 1 - [\kappa(1 - s_i) + s_i \cdot AR_i], \quad (3)$$

where $\kappa \in [0, 1]$ is a hyperparameter indicating the minimum expected AR per keypoint. To minimize the new loss, the network should predict s_i to be close to 0 if $AR_i < \kappa$ and to be near 1 conversely.

3.2 Global Descriptor Learning

As a key concept of this work, we propose to re-use the local descriptors for global retrieval. This early sharing of the computation is natural as both local

³ E.g., if the first correct correspondence appears as the 3rd nearest neighbor, then AR in the case of $k = 5$ is $(0 + 0 + 1 + 1 + 1)/5 = 0.6$.

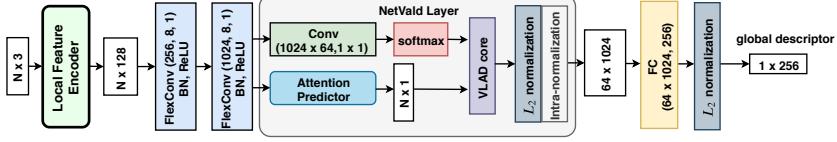


Fig. 3: The architecture of the global descriptor assembler.

and global descriptors are based on the same low-level geometric clues. The upcoming question is, how can the local descriptors be aggregated to a global one? While there exist many ways to do so, e.g., pooling, PointNet++ [39], FlexConv, PointCNN [28], Dynamic Graph CNN [59], we claim that the PCAN [65] (PointNetVLAD extended by adding attention) gives the best performance among the many and provide an ablation study in the supplementary material.

Our global aggregation network is depicted in Fig. 3. Before the NetVLAD module, two FlexConv layers are added to project the local features to a higher dimension for a more retrieval relevant encoding. The attention predictor takes these features and outputs a per-point attention map, which is followed by a NetVLAD layer to generate a compact global representation. As the output of a NetVLAD layer usually has very high dimension which indicates expensive nearest neighbor search, a FC layer is used to compress it into a lower dimension. The global descriptor assembler is trained using the same lazy quadruplet loss as used in [1, 65], to better verify our idea of using the learned local descriptor for global descriptor aggregation.

Attention map prediction. As it has been observed for image retrieval, visual cues relevant to place recognition are generally not uniformly distributed across an image. Therefore focusing on important regions is the key to improve the performance [7, 34]. However, such attention mechanism has only been explored recently for point cloud retrieval. Inspired by PCAN [65], we integrate an attention module that weighs each local descriptor before aggregation. As a key difference to PCAN, the input to our attention predictor is the learned descriptors which already encapsulate fairly good contextual information. Thus our predictor is not in charge of aggregating neighborhood information and needs a dedicated design to reflect such benefit. We thus construct our attention predictor by only chaining up three 1x1 Conv layers followed by softmax to ensure the sum of the attention weights is 1. We will show that, although our attention predictor has a much simpler structure than PCAN, yet it is effective. When combined with our descriptor, it still offers better global retrieval performance. More details on the network structure are provided in the supplementary document.

4 Experiments

The LiDAR point clouds from the Oxford RobotCar dataset [32] are used to train our network. Additionally, the ETH dataset [36] and the point clouds generated by Stereo DSO [55], a direct visual SLAM method are used to test the generalization ability of the evaluated methods. The margin used in L_{desc} is set

to $\mu = 0.5$, the minimum expected AR in Eq. 3 $\kappa = 0.6$ with $k = 5$, N in Eq. 2 is set to 512. We use a weighted sum of L_{desc} and L_{det} as the loss function to train our network $L = L_{desc} + \lambda L_{det}$.

To train the local part of our network, we use Oxford RobotCar and follow the data processing procedures in [23]. We use 35 traversals and for each create 3D submaps using the provided GPS/INS poses with a 20m trajectory and a 10m interval. The resulting submaps are downsampled using a voxel grid with grid size of 0.2m. In total 20,731 point clouds are collected for training. As the provided ground truth poses are not accurate enough to obtain cross-sequence point-to-point correspondences, we generate training samples with synthetic transformations: for a given point cloud we create another one by applying an arbitrary rotation around the upright axis and then adding Gaussian noise $\mathcal{N}(0, \sigma_{noise})$ with $\sigma_{noise} = 0.02m$. Note that as a point cloud is centered wrt. its centroid before entering the network, no translation is added to the synthetic transformations. For the global part, we use the dataset proposed in PointNetVLAD [1]. Specifically, for each of the 23 full traversals out of the 44 selected sequences from Oxford RobotCar, a testing reference map is generated consisting of the submaps extracted in the testing part of the trajectory at 20m intervals. More details on preparing the training data are left to the supplementary document.

Runtime. For a point cloud with 8192 points, our local (including feature description and keypoint detection) and global descriptors can be extracted in one forward pass in 80ms. As comparison, 3DFeatNet takes 400ms (detection)+510ms (NMS)+18ms (512 local descriptors); 3DSmoothNet needs 270ms (preprocessing)+144ms (512 local descriptors).

4.1 3D Keypoint Repeatability

We use *relative repeatability* to quantify the performance of our keypoint detector. Given two point clouds $\{\mathbf{P}, \mathbf{P}'\}$ related by a transformation T , a keypoint detector detects keypoints $K = [K_1, K_2, \dots, K_m]$ and $K' = [K'_1, K'_2, \dots, K'_m]$ from them. $K_i \in K$ is repeatable if the distance between $T(K_i)$ and its nearest neighbor $K'_j \in K'$ is less than 0.5m. *Relative repeatability* is then defined as $|K_{rep}|/|K|$ with K_{rep} the repeatable keypoints. We use the Oxford RobotCar testing set provided by 3DFeatNet [23], which contains 3426 point cloud pairs constructed from 794 point clouds. We compare to three handcrafted 3D detectors, ISS [67], SIFT-3D [29] and Harris-3D [48] and two learned ones 3DFeatNet [23] and USIP [27]. The results are presented in Fig. 4. As the most recently proposed learning based 3D detector that is dedicatedly designed for feature repeatability, USIP apparently dominates this benchmark. It is worth noting that the keypoints detected by USIP are highly clustered, which is partially in favor of achieving a high repeatability. Moreover, USIP is a pure detector, while 3DFeatNet and ours learn detection and description at the same time. Our detector outperforms all the other competitors by a large margin when detecting more than 64 keypoints. In the case of 256 keypoints, our repeatability is roughly 1.75x than the best follower 3DFeatNet. This clearly demonstrates that, when learning detector and descriptors together, *describe-and-detect* is superior than

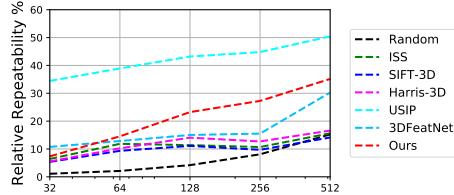


Fig. 4: Relative repeatability when different number of keypoints are detected.

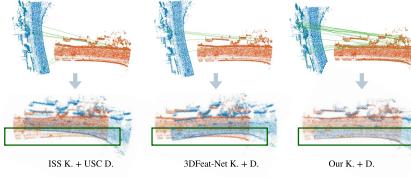


Fig. 5: Qualitative point cloud registration on Oxford RobotCar. Green lines show the inliers of RANSAC.

	RTE (m) / RRE(°) / Succ. (%) / Iter.				
	FPFH	3DSmoothNet	3DFeatNet	FCCF	DH3D
Random	0.44/1.84/89.8/7135	0.34/1.39/96.2/7274	0.43/1.62/90.5/9898	0.61/2.01/39.87/7737	0.33/1.31/92.1/6873
ISS	0.39/1.60/92.3/7171	0.32/1.21/96.8/6301	0.31/1.08/97.7/7127	0.56/1.89/43.99/7799	0.30 /1.04/97.9/4986
Harris-3D	0.54/2.31/47.5/9997	0.31/1.19/97.4/5236	0.35/1.33/95.0/9214	0.57/1.99/46.82/7636	0.34/1.20/96.4/5985
3DFeatNet	0.43/2.01/73.7/9603	0.34/1.34/95.1/7280	0.30 /1.07/98.1/2940	0.55/1.89/43.35/5958	0.32/1.24/95.4/2489
USIP	0.36/1.55/84.3/5663	0.28 / 0.93 /98.0/ 584	0.28 / 0.81 / 99.1 / 523	0.41/1.73/53.42/3678	0.30 /1.21/96.5/ 1537
DH3D	0.75/1.85/55.6/8697	0.32/1.22/96.0/3904	0.28 /1.04/ 98.2 /2908	0.38/1.48/49.47/4069	0.23 / 0.95 / 98.5 /1972

Table 1: Point cloud registration performance on Oxford RobotCar. Each row in the table corresponds to a keypoint detector and each column refers to a local 3D descriptor. In each cell, we show Relative Translational Error (RTE), Relative Rotation Error (RRE), the registration success rate and the average number of RANSAC iterations. The methods are evaluated on the testing set provided by [23]. The top three results of each metric are highlighted in **best/2nd best/3rd best**.

detect-then-describe. It is yet interesting to see how the key ideas of USIP [27] can be merged into this concept.

4.2 Point Cloud Registration

Geometric registration is used to evaluate 3D feature matching. A SE3 transformation is estimated based on the matched keypoints using RANSAC. We compute Relative Translational Error (RTE) and Relative Rotation Error (RRE) and consider a registration is successful when RTE and RRE are below 2m and 5°, respectively. We compare to two handcrafted (ISS [67] and Harris-3D [48]) and two learned (3DFeatNet [23] and USIP [27]) detectors, and three handcrafted (SI [24], USC [52] and FPFH [42]) and three learned (3DSmoothNet [18], 3DFeatNet [23] and FCCF [8]) descriptors. The average RTE, RRE, the success rate and the average number of RANSAC iterations on Oxford RobotCar of each detector-descriptor combination are shown in Tab. 1. Note that due to space limitation, only the best performing handcrafted descriptor is shown (the same applies to Tab. 2 and 4). Although USIP shows significantly better performance on repeatability, our method now delivers competitive or even better results when applied for registration, where both keypoint detector and local feature encoder are needed. This on one hand demonstrates the strong discriminative power of our local descriptors, on the other hand also supports our idea of learning detector and descriptors in the *describe-and-detect* manner. Another thing to

	RTE (m) / RRE(°) / Succ. (%) / Iter.				
	SI	3DSmoothNet	3DFeatNet	FCGF	DH3D
Random	0.36/4.36/95.2/7535	0.18 /2.73/ 100 / 986	0.30/4.06/95.2/6898	0.69/52.87/17.46/10000	0.25/3.47/ 100 /5685
ISS	0.37/5.07/93.7/7706	0.15 / 2.40 / 100 / 986	0.31/3.86/90.5/6518	0.65/24.78/6.35/10000	0.19/2.80/93.8/3635
Harris-3D	0.35/4.83/90.5/8122	0.15 /2.41/ 100 / 788	0.27/3.96/88.9/6472	0.43/55.70/6.35/10000	0.22/3.47/93.4/4524
3DFeatNet	0.35/5.77/87.3/7424	0.17 /2.73/ 100 /1795	0.33/4.50/95.2/6058	0.52/47.02/3.17/10000	0.27/3.58/93.7/6462
USIP	0.32/4.06/92.1/6900	0.18 /2.61/ 100 / 1604	0.31/3.49/82.5/7060	0.54/27.62/15.87/10000	0.29/3.29/95.2/4312
DH3D	0.42/4.65/81.3/7922	0.38/3.49/ 100 /5108	0.36/ 2.38 / 95.5 /3421	0.56/48.01/15.87/10000	0.3/ 2.02 / 95.7 /3107

Table 2: Point cloud registration performance on ETH. The top three results of each metric are highlighted in **best**/**2nd best**/**3rd best**. Note that RANSAC does not converge within the max. iterations (10000) with FCGF.

point out is that FCGF was trained on KITTI, which might explain its relatively bad results in this evaluation. Some qualitative results can be found in Fig. 5.

Unlike Oxford RobotCar, the ETH dataset [36] contains largely unstructured vegetations and much denser points, therefore is used to test the generalizability. The same detectors and descriptors as above are tested and the results are shown in Tab. 2. We notice that 3DSmoothNet shows the best performances on ETH. One important reason is that 3DSmoothNet adopts a voxel grid of size 0.02m to downsample the point clouds, while our DH3D and other methods use size 0.1m. Thus 3DSmoothNet has finer resolution and is more likely to produce smaller errors. Apart from that, our detector performs fairly well (last row) and when combine with our descriptor, it achieves the smallest rotation error. With all the detectors, FCGF descriptors cannot make RANSAC converge within the maximum number of iterations. The bad performance of FCGF is also noticed by the authors of [4] and was discussed on their GitHub page⁴.

4.3 Point Cloud Retrieval

Method	@1%	@1
PN_MAX	73.44	58.46
PN_VLAD	81.01	62.18
PCAN	83.81	69.76
Ours-4096	84.26	73.28
Ours-8192	85.30	74.16

Table 3: Average recall (%) at top 1% and top 1 for Oxford RobotCar.

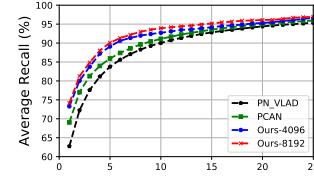


Fig. 6: Average recall of the top 25 retrievals on Oxford RobotCar.

We compare our method against the two state-of-the-art approaches, PCAN [65] and PointNetVLAD (PN_VLAD) [1]. We also report the results of PN_MAX presented in PN_VLAD, which consists of the original PointNet architecture with the maxpool layer and a fully connected layer to produce a global descriptor. Note that both PN_VLAD and PCAN take submaps of size 4096, whereas ours takes 8192 to favor local feature extraction. We thus add a downsampling layer before the final NetVLAD layer to make sure the same size of 4096 points enter

⁴ <https://github.com/XuyangBai/D3Feat/issues/1>

the final aggregation procedure. For demonstration, we also report the results of using the default setting. We first evaluate the average recall at top 1% and top1 and show the results in Tab. 3. Our method with both settings outperforms all the other methods. We further show the recall curves of the top25 retrieval matches in Fig. 6, where both of our networks consistently outperform the other two state-of-the-art approaches. The evaluation results prove that our network can effectively take advantage of the informative local features and produce more discriminative global descriptors. Also note that even with the number of points halved before entering the NetVLAD layer, the performance drop of our method is very small, which shows that our local features integrate sufficient contextual information for global retrieval. Some qualitative retrieval results are provided in the supplementary document.

4.4 Application to Visual SLAM

	RTE (m) / RRE($^{\circ}$) / Succ. (%) / Iter.				
	FPFH	3DSmoothNet	3DFeatNet	FCGF	DH3D
Random	0.56/2.82/53.13/9030	0.70/2.19/73.1/6109	0.72/2.37/69.0/9661	0.51/2.65/74.93/5613	0.70/2.23/71.9/7565
ISS	0.56/3.03/43.58/9210	0.67/2.15/79.1/6446	0.58/2.41/71.9/9776	0.51/2.57/71.94/6015	0.48/ 1.72 / 90.2 /6312
Harris-3D	0.49/2.67/45.67/9130	0.48/2.07/74.9/6251	0.66/2.26/64.5/9528	0.48/2.63/74.03/5482	0.39/2.27/68.1/7860
3DFeatNet	0.62/3.05/35.52/7704	0.38 /2.22/66.6/5235	0.92/1.97/84.1/8071	0.54/2.64/60.90/ 4409	0.74/2.38/80.9/7124
USIP	0.54/2.98/48.96/7248	0.39/2.27/77.3/5593	0.85/2.24/69.9/8389	0.51/2.65/67.46/ 3846	0.65/2.45/68.1/6824
DH3D	0.60/2.92/48.96/8914	0.35 /2.01/77.9/5764	0.41/ 1.84 / 89.3 /7818	0.48/2.43/69.55/ 5002	0.36 / 1.58 / 90.6 /7071

Table 4: Generalization of point cloud registration for visual SLAM. In this experiment, point clouds are generated by running Stereo DSO [55] on Oxford. For learning based methods, models trained on LiDAR points are used without fine-tuning. The top three results of each metric are highlighted in **best/2nd best/3rd best**.

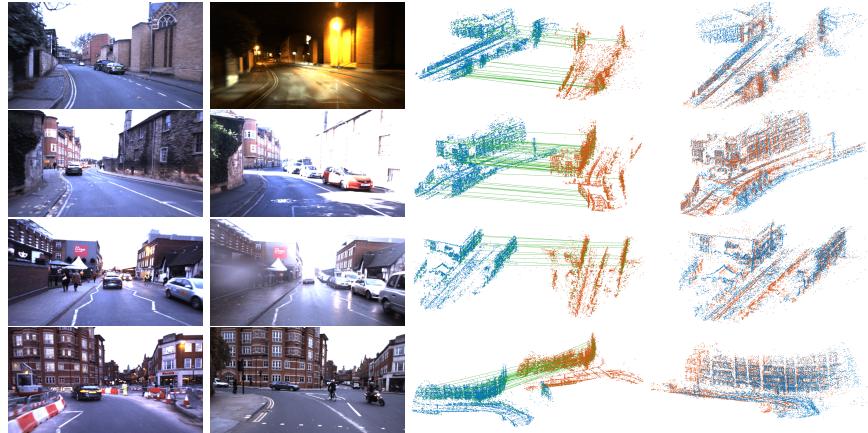


Fig. 7: Registration on point clouds generated by Stereo DSO [55]. The first two columns display frames from the reference and the query sequences. The last two columns show the matches found by RANSAC and the point clouds after alignment.

In this section, we demonstrate the generalization capability of our method to a different sensor modality by evaluating its performance on the point clouds generated by Stereo DSO [55]. As a direct method, Stereo DSO has the advantage of delivering relatively dense 3D reconstructions, which can provide stable geometric structures that are less affected by image appearance changes. We therefore believe that it is worth exploring extracting 3D descriptors from such reconstructions which can be helpful for loop-closure and relocalization for visual SLAM. To justify this idea, Stereo DSO is used to generate point clouds of eight traversals from Oxford RobotCar, which covers a wide range of day-time and weather conditions. This gives us 318 point cloud pairs with manually annotated relative poses. Each point cloud is cropped with a radius of 30m and randomly rotated around the vertical axis. We use the same parameters as in Sec. 4.2 without fine-tuning our network and evaluate the geometric registration performance against other methods in Tab. 4. As shown in the table, our approach achieves the best rotation error (1.58°) and success rate (90.6%) and second best translation error (0.36m) among all the evaluated methods. It can also be noticed that most evaluated methods show significant inferior performances compared to the results in Tab. 1, e.g., the successful rates of 3DFeatNet+3DFeatNet, USIP+3DSmoothNet and USIP+3DFeatNet drop from 98.1%, 98.0% and 99.1% to 84.1%, 77.3% and 69.9%, respectively. This is largely because the point clouds extracted from LiDAR scannings have quite different distributions as those from Stereo DSO. Our model is still able to achieve a successful rate of 90.6%, showing the least degree of degeneracy. This further demonstrates the good generalization ability of the proposed method. Some qualitative results are shown in Fig. 7.

4.5 Ablation Study

Effectiveness of different components. We carry out three experiments to explore the contributions of different components of our method: (1) We remove the detector module and only L_{desc} is used to train the local feature encoder; (2) The weak supervision at the submap level proposed by 3DFeatNet [23] is used (details in the supplementary material); (3) We remove the SE blocks. As shown in Tab. 5, the largest performance decrease comes with (2), which verifies our idea of generating a supervision signal by synthesizing transformations. Results of (1) indicate that learning an effective confidence map S can improve the quality of the learned local descriptors for matching. The results of (3) show that SE blocks contribute to learning more informative local descriptors and therefore are helpful to 3D feature matching.

Robustness test. We assess the robustness of our model for both point cloud retrieval and registration against three factors, i.e., noise, rotation and down-sampling: We add Gaussian noise $\mathcal{N}(0, \sigma_{noise})$ to the point clouds; The range of the rotation test is set between 0 and 90° ; Point clouds are downsampled using a set of factors α . For the local part, as shown in Fig. 8, our descriptors has shown excellent rotation invariance. When noise is added, our method can still achieves $> 90\%$ success rate for $\sigma_{noise} < 0.15m$. The performance significantly drops for

Method	RTE(m)	RRE($^{\circ}$)	Succ.	Iter.
w/o L_{det}	0.43	1.52	93.72	3713
Weak Sup.	0.48	1.78	90.82	3922
w/o SE	0.39	1.24	95.18	3628
Default	0.23	0.95	98.49	1972

Table 5: Effects of different components for point cloud registration on Oxford RobotCar.

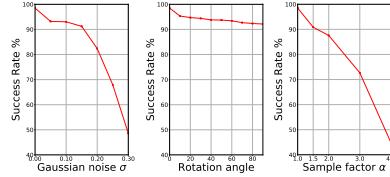


Fig. 8: Local detector and descriptor robustness test evaluated by the success rate of point cloud registration.

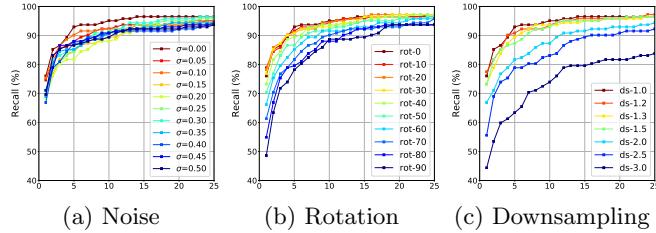


Fig. 9: Global descriptor robustness against random noise, rotation and downsampling. The x axes show the number of top retrieved matches.

$\sigma_{noise} > 0.2m$, possibly due to the fact that training samples are filtered by a voxel grid with size $0.2m$, thus strong noise tends to heavily change the underlying point distribution. A similar explanation applies to the case of downsampling for a factor $\alpha > 2$. Nevertheless, our model can still guarantee 90% success rate for $\alpha \leq 1.5$. We conduct the same robustness tests for our global descriptors. Fig. 9 (a) demonstrates that our global descriptors possess good robustness to noise. Contrary to the local descriptors, the global descriptor seems to be less robust against rotations, which needs further investigation. Similar to the local descriptor, the quality of global feature is not affected too much for $\alpha \leq 1.5$.

5 Conclusion

We introduced a hierarchical network for the task of large-scale point cloud based relocalization. Rather than pursuing the traditional strategy of detect-then-describe or separately computing local and global descriptors, our network performs local feature detection, local feature description and global descriptor extraction in a single forward pass. Experimental results demonstrate the state-of-the-art performance of both our local and global descriptors across multiple benchmarks. Our model trained on LiDAR points also shows favorable generalization ability when applied to point clouds generated by visual SLAM methods. Future work is focused on further exploring the robustness of the learned descriptors to various perturbations.

References

1. Angelina Uy, M., Hee Lee, G.: PointNetVLAD: Deep point cloud based retrieval for large-scale place recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4470–4479 (2018)
2. Arandjelovic, R., Gronat, P., Torii, A., Pajdla, T., Sivic, J.: NetVLAD: CNN architecture for weakly supervised place recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5297–5307 (2016)
3. Axelrod, B., Kaelbling, L.P., Lozano-Pérez, T.: Provably safe robot navigation with obstacle uncertainty. *The International Journal of Robotics Research* **37**(13-14), 1760–1774 (2018)
4. Bai, X., Luo, Z., Zhou, L., Fu, H., Quan, L., Tai, C.L.: D3Feat: Joint Learning of Dense Detection and Description of 3D Local Features. arXiv:2003.03164 [cs.CV] (2020)
5. Cao, F., Zhuang, Y., Zhang, H., Wang, W.: Robust place recognition and loop closing in laser-based SLAM for UGVs in urban environments. *IEEE Sensors Journal* **18**(10), 4242–4252 (2018)
6. Chen, Z., Jacobson, A., Sünderhauf, N., Upcroft, B., Liu, L., Shen, C., Reid, I., Milford, M.: Deep learning features at scale for visual place recognition. In: 2017 IEEE International Conference on Robotics and Automation (ICRA). pp. 3223–3230. IEEE (2017)
7. Chen, Z., Liu, L., Sa, I., Ge, Z., Chli, M.: Learning context flexible attention model for long-term visual place recognition. *IEEE Robotics and Automation Letters* **3**(4), 4015–4022 (2018)
8. Choy, C., Park, J., Koltun, V.: Fully convolutional geometric features. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 8958–8966 (2019)
9. Cop, K.P., Borges, P.V., Dubé, R.: DELIGHT: An efficient descriptor for global localisation using LiDAR intensities. In: 2018 IEEE International Conference on Robotics and Automation (ICRA). pp. 3653–3660. IEEE (2018)
10. Deng, H., Birdal, T., Ilic, S.: PPF-FoldNet: Unsupervised learning of rotation invariant 3D local descriptors. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 602–618 (2018)
11. Deng, H., Birdal, T., Ilic, S.: PPFNet: Global context aware local features for robust 3D point matching. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 195–205 (2018)
12. Deng, H., Birdal, T., Ilic, S.: 3D local features for direct pairwise registration. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3244–3253 (2019)
13. Deschaud, J.E.: IMLS-SLAM: scan-to-model matching based on 3D data. In: 2018 IEEE International Conference on Robotics and Automation (ICRA). pp. 2480–2485. IEEE (2018)
14. DeTone, D., Malisiewicz, T., Rabinovich, A.: SuperPoint: Self-supervised interest point detection and description. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. pp. 224–236 (2018)
15. Dusmanu, M., Rocco, I., Pajdla, T., Pollefeys, M., Sivic, J., Torii, A., Sattler, T.: D2-Net: A trainable cnn for joint description and detection of local features. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 8092–8101 (2019)

16. Elbaz, G., Avraham, T., Fischer, A.: 3D point cloud registration for localization using a deep neural network auto-encoder. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4631–4640 (2017)
17. Engel, J., Stückler, J., Cremers, D.: Large-scale direct SLAM with stereo cameras. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 1935–1942. IEEE (2015)
18. Gojcic, Z., Zhou, C., Wegner, J.D., Wieser, A.: The Perfect Match: 3D point cloud matching with smoothed densities. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5545–5554 (2019)
19. Gordo, A., Almazán, J., Revaud, J., Larlus, D.: Deep image retrieval: Learning global representations for image search. In: European conference on computer vision. pp. 241–257. Springer (2016)
20. Granström, K., Schön, T.B., Nieto, J.I., Ramos, F.T.: Learning to close loops from range data. *The International Journal of Robotics Research* **30**(14), 1728–1754 (2011)
21. Groh, F., Wieschollek, P., Lensch, H.P.: Flex-convolution. In: Asian Conference on Computer Vision. pp. 105–122. Springer (2018)
22. Hu, J., Shen, L., Sun, G.: Squeeze-and-Excitation networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7132–7141 (2018)
23. Jian Yew, Z., Hee Lee, G.: 3DFeat-Net: Weakly supervised local 3D features for point cloud registration. In: Proceedings of the European Conference on Computer Vision. pp. 607–623 (2018)
24. Johnson, A.E.: Spin-Images: A representation for 3D surface matching (1997)
25. Khouri, M., Zhou, Q.Y., Koltun, V.: Learning compact geometric features. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 153–161 (2017)
26. Kim, G., Kim, A.: Scan context: Egocentric spatial descriptor for place recognition within 3D point cloud map. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 4802–4809. IEEE (2018)
27. Li, J., Lee, G.H.: USIP: Unsupervised stable interest point detection from 3D point clouds. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 361–370 (2019)
28. Li, Y., Bu, R., Sun, M., Wu, W., Di, X., Chen, B.: PointCNN: Convolution on x -transformed points. In: Advances in Neural Information Processing Systems. pp. 820–830 (2018)
29. Lowe, D.G.: Local feature view clustering for 3D object recognition. In: Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. vol. 1, pp. I–I. IEEE (2001)
30. Lu, W., Wan, G., Zhou, Y., Fu, X., Yuan, P., Song, S.: DeepVCP: An end-to-end deep neural network for point cloud registration. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 12–21 (2019)
31. Lu, W., Zhou, Y., Wan, G., Hou, S., Song, S.: L3-Net: Towards learning based lidar localization for autonomous driving. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6389–6398 (2019)
32. Maddern, W., Pascoe, G., Linegar, C., Newman, P.: 1 year, 1000 km: The Oxford RobotCar dataset. *The International Journal of Robotics Research* **36**(1), 3–15 (2017)
33. Mur-Artal, R., Montiel, J.M.M., Tardos, J.D.: ORB-SLAM: A versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics* **31**(5), 1147–1163 (2015)

34. Noh, H., Araujo, A., Sim, J., Weyand, T., Han, B.: Large-scale image retrieval with attentive deep local features. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 3456–3465 (2017)
35. Ort, T., Paull, L., Rus, D.: Autonomous vehicle navigation in rural environments without detailed prior maps. In: 2018 IEEE International Conference on Robotics and Automation (ICRA). pp. 2040–2047. IEEE (2018)
36. Pomerleau, F., Liu, M., Colas, F., Siegwart, R.: Challenging data sets for point cloud registration algorithms. *The International Journal of Robotics Research* **31**(14), 1705–1711 (2012)
37. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: PointNet: Deep learning on point sets for 3D classification and segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 652–660 (2017)
38. Qi, C.R., Su, H., Nießner, M., Dai, A., Yan, M., Guibas, L.J.: Volumetric and multi-view CNNs for object classification on 3D data. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 5648–5656 (2016)
39. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: PointNet++: Deep hierarchical feature learning on point sets in a metric space. In: Advances in Neural Information Processing Systems. pp. 5099–5108 (2017)
40. Revaud, J., Weinzaepfel, P., de Souza, C.R., Humenberger, M.: R2D2: repeatable and reliable detector and descriptor. In: NeurIPS (2019)
41. Röhling, T., Mack, J., Schulz, D.: A fast histogram-based similarity measure for detecting loop closures in 3-D LiDAR data. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 736–741. IEEE (2015)
42. Rusu, R.B., Bradski, G., Thibaux, R., Hsu, J.: Fast 3D recognition and pose using the viewpoint feature histogram. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 2155–2162. IEEE (2010)
43. Rusu, R.B., Marton, Z.C., Blodow, N., Beetz, M.: Persistent point feature histograms for 3D point clouds. In: Proc 10th Int Conf Intel Autonomous Syst (IAS-10), Baden-Baden, Germany. pp. 119–128 (2008)
44. Sarlin, P.E., Cadena, C., Siegwart, R., Dymczyk, M.: From coarse to fine: Robust hierarchical localization at large scale. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 12716–12725 (2019)
45. Sarlin, P.E., Debraine, F., Dymczyk, M., Siegwart, R., Cadena, C.: Leveraging deep visual descriptors for hierarchical efficient localization. arXiv preprint arXiv:1809.01019 (2018)
46. Sattler, T., Torii, A., Sivic, J., Pollefeys, M., Taira, H., Okutomi, M., Pajdla, T.: Are large-scale 3D models really necessary for accurate visual localization? In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1637–1646 (2017)
47. Simonovsky, M., Komodakis, N.: Dynamic edge-conditioned filters in convolutional neural networks on graphs. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3693–3702 (2017)
48. Sipiran, I., Bustos, B.: Harris 3D: a robust extension of the Harris operator for interest point detection on 3D meshes. *The Visual Computer* **27**(11), 963 (2011)
49. Su, H., Jampani, V., Sun, D., Maji, S., Kalogerakis, E., Yang, M.H., Kautz, J.: SplatNet: Sparse lattice networks for point cloud processing. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2530–2539 (2018)
50. Su, H., Maji, S., Kalogerakis, E., Learned-Miller, E.: Multi-view convolutional neural networks for 3D shape recognition. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 945–953 (2015)

51. Taira, H., Okutomi, M., Sattler, T., Cimpoi, M., Pollefeys, M., Sivic, J., Pajdla, T., Torii, A.: InLoc: Indoor visual localization with dense matching and view synthesis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7199–7209 (2018)
52. Tomboli, F., Salti, S., Di Stefano, L.: Unique shape context for 3D data description. In: Proceedings of the ACM Workshop on 3D Object Retrieval. pp. 57–62. ACM (2010)
53. Wang, P., Yang, R., Cao, B., Xu, W., Lin, Y.: Dels-3D: Deep localization and segmentation with a 3D semantic map. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5860–5869 (2018)
54. Wang, P.S., Liu, Y., Guo, Y.X., Sun, C.Y., Tong, X.: O-CNN: Octree-based convolutional neural networks for 3D shape analysis. ACM Transactions on Graphics (TOG) **36**(4), 72 (2017)
55. Wang, R., Schworer, M., Cremers, D.: Stereo DSO: Large-scale direct sparse visual odometry with stereo cameras. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 3903–3911 (2017)
56. Wang, S., Suo, S., Ma, W.C., Pokrovsky, A., Urtasun, R.: Deep parametric continuous convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2589–2597 (2018)
57. Wang, W., Yu, R., Huang, Q., Neumann, U.: SGPN: Similarity group proposal network for 3D point cloud instance segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2569–2578 (2018)
58. Wang, Y., Solomon, J.M.: Deep Closest Point: Learning representations for point cloud registration. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 3523–3532 (2019)
59. Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M.: Dynamic graph CNN for learning on point clouds. ACM Transactions on Graphics (TOG) **38**(5), 1–12 (2019)
60. Yi, K.M., Trulls, E., Lepetit, V., Fua, P.: LIFT: Learned invariant feature transform. In: European Conference on Computer Vision. pp. 467–483. Springer (2016)
61. Yin, H., Wang, Y., Tang, L., Ding, X., Xiong, R.: LocNet: Global localization in 3D point clouds for mobile robots. In: Proceedings of the 2018 IEEE Intelligent Vehicles Symposium (IV), Changshu, China. pp. 26–30 (2018)
62. Zeng, A., Song, S., Nießner, M., Fisher, M., Xiao, J., Funkhouser, T.: 3DMatch: Learning local geometric descriptors from RGB-D reconstructions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1802–1811 (2017)
63. Zhang, J., Singh, S.: LOAM: Lidar odometry and mapping in real-time. In: Robotics: Science and Systems Conference (RSS). Berkeley, CA (July 2014)
64. Zhang, J., Singh, S.: Visual-lidar odometry and mapping: Low-drift, robust, and fast. In: 2015 IEEE International Conference on Robotics and Automation (ICRA). pp. 2174–2181. IEEE (2015)
65. Zhang, W., Xiao, C.: PCAN: 3D attention map learning using contextual information for point cloud based retrieval. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 12436–12445 (2019)
66. Zhao, Y., Birdal, T., Deng, H., Tomboli, F.: 3D point capsule networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1009–1018 (2019)
67. Zhong, Y.: Intrinsic shape signatures: A shape descriptor for 3D object recognition. In: 2009 IEEE 12th International Conference on Computer Vision Workshops. pp. 689–696. IEEE (2009)

68. Zhou, Y., Tuzel, O.: Voxelnet: End-to-end learning for point cloud based 3d object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4490–4499 (2018)

Supplementary Material

DH3D: Deep Hierarchical 3D Descriptors for Robust Large-Scale 6DoF Relocalization

Juan Du^{1*}, Rui Wang^{1,2*}, and Daniel Cremers^{1,2}

¹ Technical University of Munich

² Artisense

`{duj, wangr, cremers}@in.tum.de`

1 Overview

Despite the significant progresses in deep learning, one issue we are commonly facing is that reproducibility is not guaranteed by all the papers. In this respect, in addition to releasing our code, we provide in this supplementary document the remaining technical aspects and some insights we have gathered during the development. We hope this can help the other researchers in this field.

In the first part, we show more detailed results of our keypoint repeatability on the ETH dataset and on the point clouds generated by Stereo DSO on Oxford RobotCar (referred to as StereoDSO-Oxford), which we omitted in the main paper due to space limitation. We also provide more qualitative registration results on Oxford RobotCar and StereoDSO-Oxford, including some failure cases. The second part provides results revealing the influence of using different operators for global aggregation, where we see that although there exist several ways, attention based NetVLAD layer is so far the best choice. Some qualitative results on global retrieval are attached afterwards. The third part is dedicated to presenting all the technical details left over from the main paper, including (1) structures of sub-networks, (2) training data preparation, (3) two-phase training of the local and global networks, (4) generation of the Stereo DSO point clouds, and (5) details on the weak supervision used in the ablation study in the main paper.

2 Additional Results on Local Feature

2.1 Keypoint Repeatability

We use the same method as in the main paper to evaluate keypoint repeatability on ETH and StereoDSO-Oxford, except that now we choose 0.3m as the threshold distance for the ETH dataset to determine whether a point is repeatable. For all the detectors, 512 and 1024 keypoints are respectively extracted for StereoDSO-Oxford and ETH. The results are shown in Fig. 1.

* Authors contributed equally.

On StereoDSO-Oxford, the results hold a similar pattern as those on the Oxford RobotCar LiDAR points shown in the main paper. Our detector outperforms other methods except USIP which is a pure detector that dedicatedly designed for feature repeatability. All the handcrafted detectors and 3DFeatNet have repeatability lower than 0.2. On the ETH dataset, our detector has the highest repeatability while the performance of USIP degrades significantly. When USIP is trained, a relatively fixed receptive field is defined for extracting each feature location (by pre-setting the parameter M and K , please refer to [4] for details). In other words, the performance of the learned network is highly correlated to certain point densities and scales in the receptive field. Therefore, when applied to point clouds with very different spatial distribution as the training data, the network suffers to generalize well.

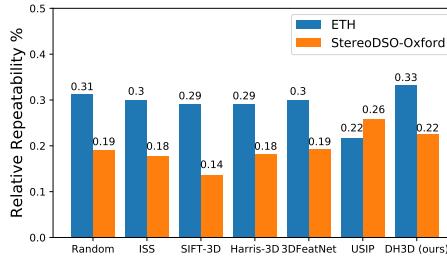


Fig. 1: Relative repeatability on ETH dataset and StereoDSO-Oxford testing set.

2.2 Qualitative Results on Oxford RobotCar

We show additional qualitative registration results by our method in Fig. 2, including two failure cases due to the domination of local features from the facade of buildings which are repeated and contain limited local textures.

2.3 Qualitative Results on Stereo DSO Points

Fig. 3 shows more registration results of StereoDSO-Oxford, which covers different conditions including night, dusk, direct sun, snow, rain and roadworks. Compared to LiDAR scanner, due to the nature of tending to reconstructed 3D points corresponding to pixels with high image gradients, visual SLAM methods usually generate point clouds with evidently different spatial densities and distributions under different lighting conditions. Such variations become even significant when further combined with different scene layouts. This can be clearly observed in the figure. Still, our keypoint detector and local descriptors trained on LiDAR points are able to achieve fairly good matchings without fine-tuning.

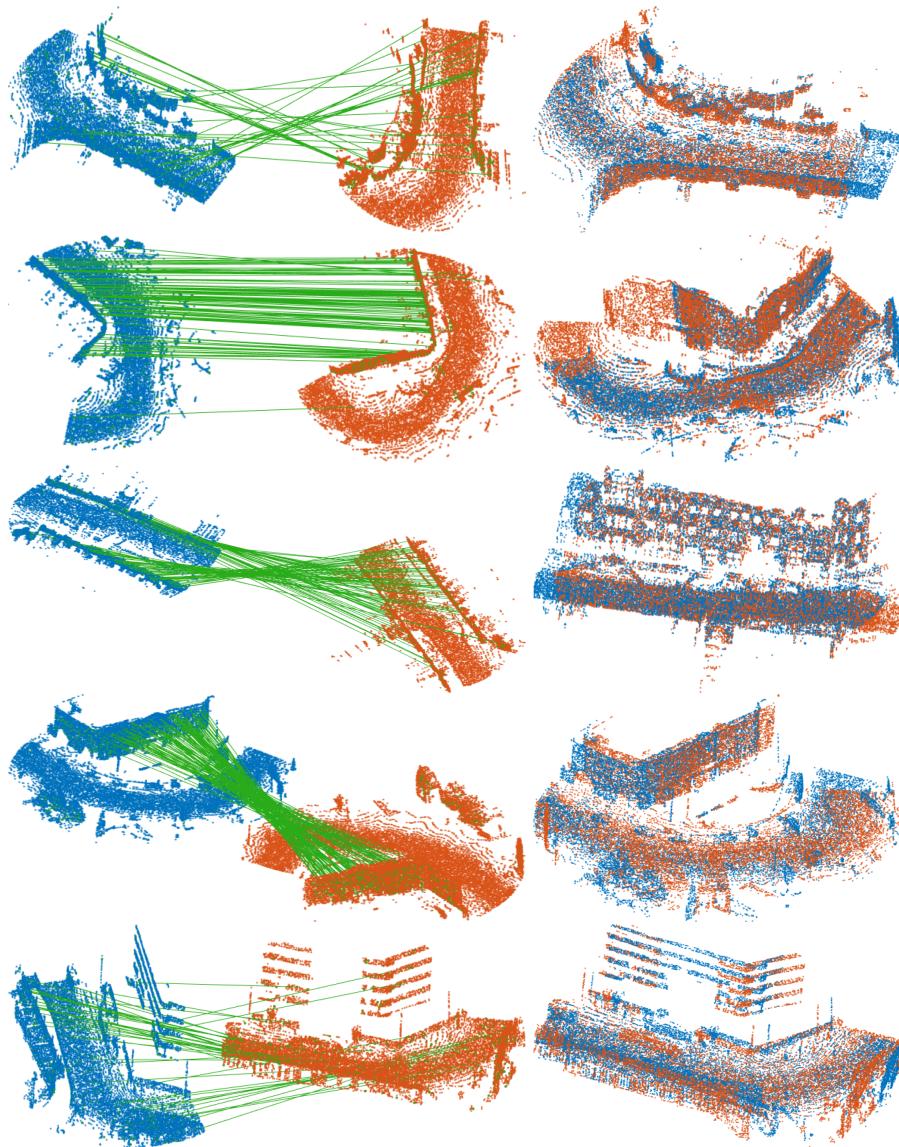


Fig. 2: More results of registration of Oxford RobotCar. The first column shows the matched features (after RANSAC) and the second shows the point clouds after alignment. The last two rows show failure cases caused by the dominated repetitive vertical structures.

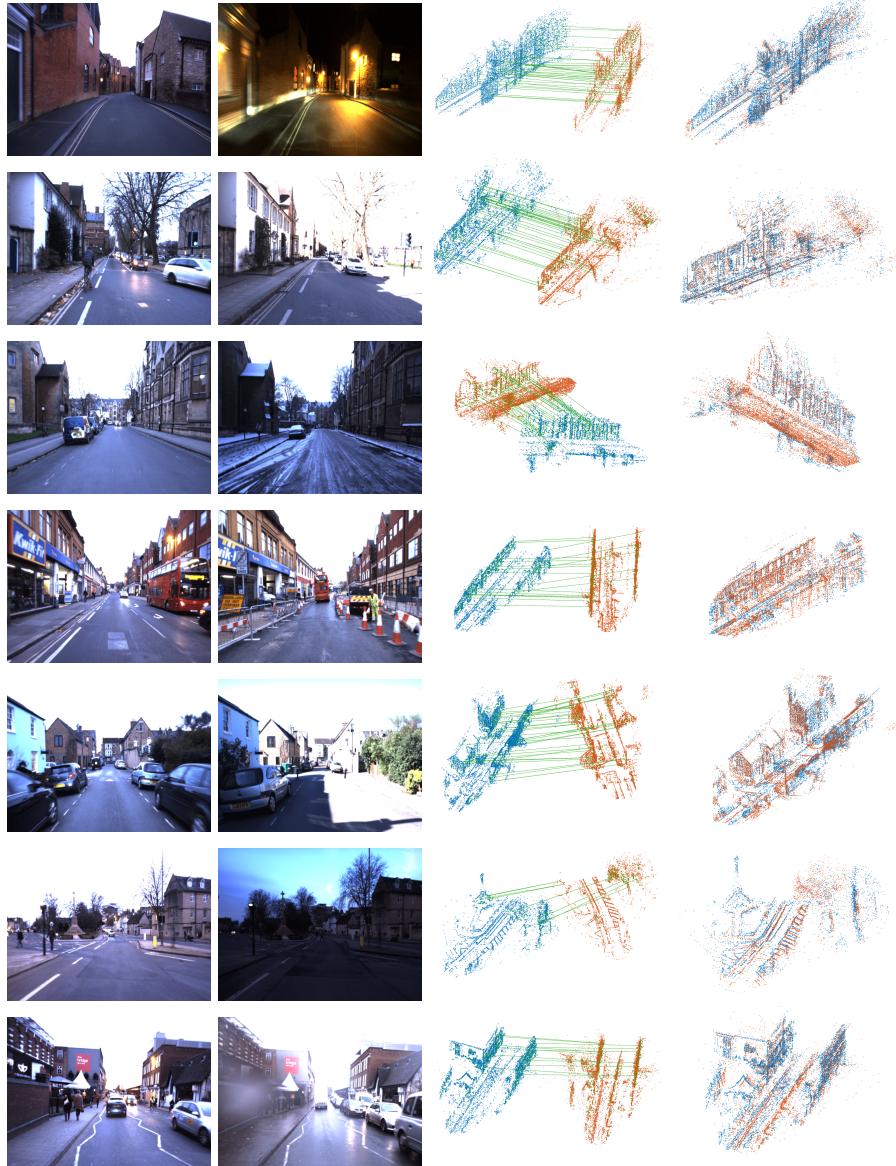


Fig. 3: More results of registration of StereoDSO-Oxford. The first two columns display frames from the reference and the query sequences. The last two columns show the matched features (after RANSAC) and the point clouds after alignment. Note that under different lighting and weather conditions, the point clouds generated by Stereo DSO have very different spatial densities and distributions.

3 Additional Results on Global Descriptor

3.1 Comparison of Different Global Aggregators

As mentioned in the main paper, there exist many ways to aggregate the extracted local descriptors into a global one. Apart from the attention based NetVLAD model, which is adopted as our global assembler, we tested four other models, i.e. PointNet++ [7], Dynamic Graph CNN (DGCNN) [9], PointCNN [5], and FlexConv [2]. Max- and average-pooling (Max-Pool, Avg-Pool) are tested additionally as two baselines. We first explain the details on each model and discuss the results presented in Fig. 4 afterwards.

PoinNet++: To construct a PoinNet++ style global assembler, we use a combination of Multi-Scale Grouping (MSG) layer and a Set Abstraction(SA) layer proposed in [7]. The architecture is as follows: SA(1024, [4.0, 8.0], [16, 32], [[128, 128, 256], [128, 128, 256]]) → SA([256, 512, 1024]) → FC(512, 0.5) → FC(256, 0.5), where $\text{SA}(K, r, [l_1, \dots, l_d])$ is a SA level with K local regions of ball radius r and use a PointNet of d fully connected layers with width $l_i (i = 1, \dots, d)$. Note that all the models discussed in this section include a final fully connected layer to transform the global descriptor to a fixed dimension of 256.

DGCNN: We use two consecutive EdgeConv blocks (MLP(128, 128) and MLP(256, 256)) to process the local descriptors, where $\text{MLP}(a_1, \dots, a_n)$ is a multi-layer perceptron with the number of layer neurons defined as (a_1, a_2, \dots, a_n) . The output of these two layers are concated which is followed by another MLP(1024) layer to generate a tensor of shape $N \times 1024$. Then, a global max pooling is used to get the global descriptor.

PointCNN: Inspired by PointCNN, we also test the idea of incorporating χ -convolution to extract a global descriptor. A χ -conv(N, C, K, D) layer takes N points as input and outputs a tensor of shape $N \times C$. It is defined in a local region which constructed by sampling K input points from $K \times D$ neighboring points, where D is the dilation rate. In our settings, we add four χ -conv layers to transform the local features as follows: $\chi\text{-conv}(2048, 256, 16, 2) \rightarrow \chi\text{-conv}(768, 512, 16, 2) \rightarrow \chi\text{-conv}(384, 512, 16, 2) \rightarrow \chi\text{-conv}(128, 1024, 16, 2)$.

FlexConv: We add four additional FlexConv layers as follows, FlexConv(256, 8, 512) → FlexConv(256, 8, 128) → FlexConv(512, 8, 32) → FlexConv(1024, 8, 32), where $\text{FlexConv}(D, k, N)$ denotes a Flex-Convolution operation on N input points with the neighborhood size as k and D is the dimension of the output. Then, we apply a Flex-Convolution at the center point $(0, 0, 0)$ with all the remaining points as neighbors to generate a global descriptor.

Pooling: The Max-Pool and Avg-Pool take the extracted local feature map as input and perform a simple max/avg pooling to produce a single global embedding.

Fig.4 shows the average recall curves within the top 25 retrievals on the Oxford RobotCar dataset. We draw the overall conclusion from these results that the

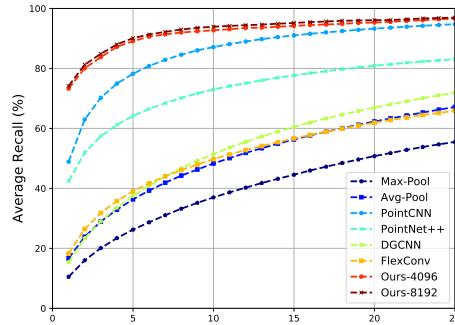


Fig. 4: Average recall of the top 25 retrievals of different approaches for assembling global descriptors on the Oxford RobotCar dataset.

attention based NetVLAD global aggregation is so far the best option among the multiple tested models. This validates our choice on the global aggregator in the main paper. In addition, we can observe that among all the baseline methods, PointCNN far outperforms others. One possible explanation is that PointCNN estimates a transformation on the input points which sufficiently exploits the 3D position information compared to other methods. This can also account for the second highest recall obtained by PointNet++ which takes advantage of the 3D position implicitly by using an MSG layer. FlexConv, DGCNN and Avg-Pool produce similar results while Max-Pool gives the worst performance, most likely because a simple max pooling causes a large amount of loss of information.

3.2 Quanlitative Results on Point Cloud Retrieval

In addition to the quantitative results of our global descriptors (Ours-8192) on point cloud retrieval reported in Sec. 4.3 of the main paper, we provide some qualitative examples in Fig. 5. To this end, we first randomly select a full traversal (Sequence 2014-12-10-18-10-50) as the reference map. Then three query point clouds from three other randomly selected traversals in the remaining 32 sequences are chosen, each representing one sample submap from individual testing areas as shown in the right sub-figure of Fig. 8. For each example, we display the query point cloud and the top5 retrieved results. In addition, the location of each point cloud is plotted in the reference map on the right. We use different colors to indicate the L2-distances between the global descriptors of the query point clouds and those of the submaps contained in the reference map. For each query, the best match (represented by the blue square) correctly overlaps with the target location (indicated by the red circle), which demonstrates that our global descriptors have good robustness to rotation and occlusion. It is also worth noting that most of the un-retrieved submaps in the reference map have relatively high distances in the descriptor field, demonstrating the good discriminativeness of our global descriptor.

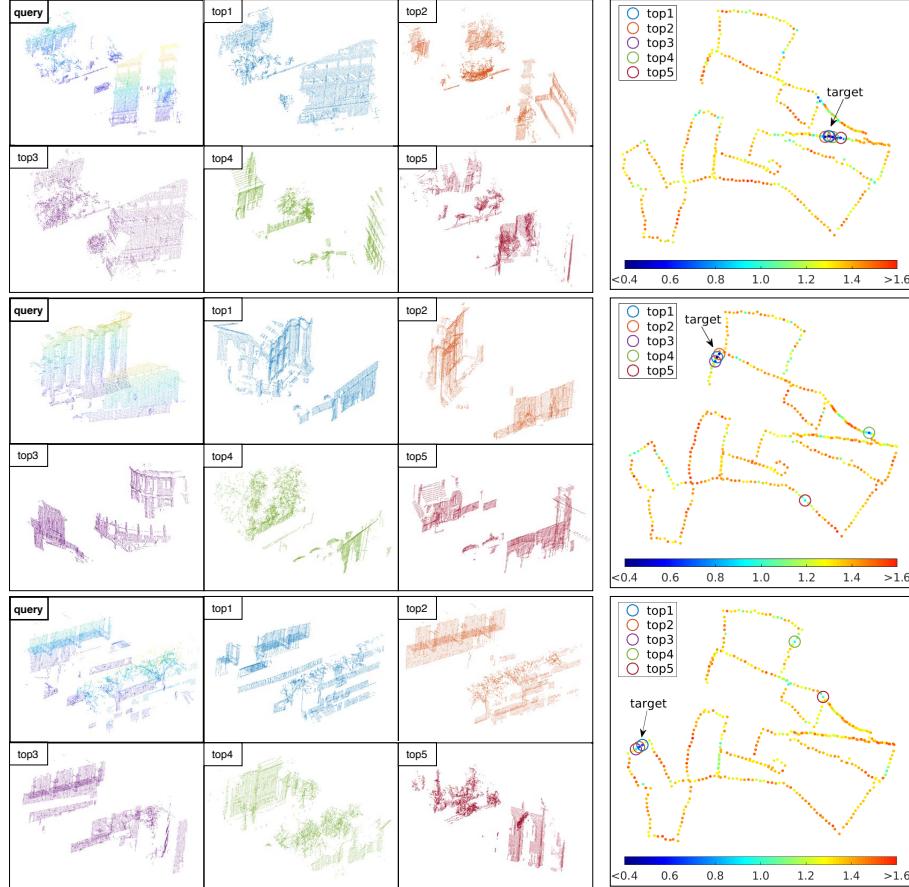


Fig. 5: Visualizations of example retrieval results of our network on the Oxford Robot-Car dataset. For each retrieval, we display the query point cloud and the top5 matches returned by our method. We also indicate the 3D locations of these point clouds in the associated reference map. Colors in each query point cloud indicate different heights above ground while colors of the retrieved point clouds correspond to the markers drawn in the reference maps. The L2-distances between the global descriptor of the query point cloud and those of all the submaps are color-coded.

4 Technical Details

4.1 Additional Network Details

The architectures of our local feature encoder and global descriptor assembler are illustrated in the main paper. Here we provide the structural details of the other two sub-networks, i.e., the keypoint detector and the attention map predictor.

3D Keypoint detector. Fig. 6 depicts the architecture of our 3D keypoint detector. As explained in the main paper, it consists of 4 1×1 convolution layers and a sigmoid activation function at the end. All the convolution layers are followed by ReLU non-linear activation and BN (BatchNorm). The 1×1 convolutions essentially transform each point independently and are invariant to point ordering. The sigmoid activation function is used to restrict the individual output value to $[0, 1]$, which represents the matching reliability of each local descriptor.

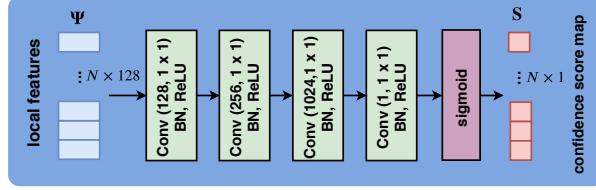


Fig. 6: The architecture of the local feature detector in Fig. 2 of the main paper.

Attention map predictor. We use a sub-network which has similar structure as the detector to predict attentive weight for each local descriptor. The main component is composed of 3 1×1 convolution layers with ReLU activation and BN. A softmax operation is applied to produce the final attention distribution that sums up to 1. Recall that in our case, the inputs to the attention predictor are the learned local descriptors, which have already encoded multi-level contextual information. Thus our attention predictor can benefit from this and have a relatively simple yet effective model structure.

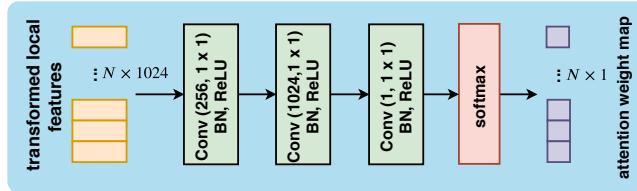


Fig. 7: The architecture of the attention predictor in Fig. 3 of the main paper.

4.2 Training Data Preparation

Data splitting. We use the LiDAR points from the Oxford RobotCar dataset [6] to learn both local and global descriptors. For fair comparisons with other methods, we split the dataset differently for these two tasks when constructing the training and testing data. More specifically, for the keypoint detection and local descriptor extraction, we follow the practice of 3DFeatNet [3] and make use of 40 full traversals with each traversal split into two disjoint sets for training and testing. The training set is constructed by the point clouds accumulated from the training region of the first 35 sequences. The point clouds collected from the testing region of the remaining 5 sequences are used to generate the testing set. For global descriptor learning, we adopt the split used in PointNetVLAD [1], where a set of 44 traversals are used, both full and partial. Each run is then geographically split into 70% and 30% for training and testing, respectively. A comparison of the above two data splittings is illustrated in Fig. 8.

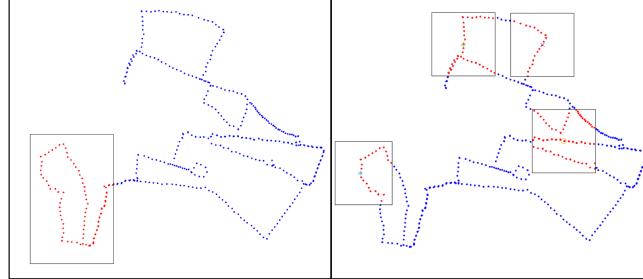


Fig. 8: Different data splitting strategies for learning local descriptors and keypoint detector (left) and global descriptors (right). Blue points represent submaps in the training set and red points represent those in the testing set.

Data preprocessing. For each traversal, we create a 3D point cloud submap with a 20m radius for every 10m interval whenever good GPS/INS poses are available. According to the widely adopted convention, the ground planes are removed in all submaps since they are repetitive structures and non-informative. The resulting submaps are then downsampled using a VoxelGrid filter with a grid size of 0.2m. For each point cloud, 8192 points are randomly chosen on the fly during training. Adopting the data splitting strategies as explained above, we obtain 20,731 point clouds for learning local descriptor and detector and 21,026 submaps for training global descriptor. As mentioned in the main paper, due to the lack of accurate point-to-point correspondences between different sequences, we generate training samples for local descriptor and detector learning with synthetic transformations (arbitrary rotations around the upright axis and Gaussian noise with $\sigma_{noise} = 0.02m$). The correspondence matrix $\mathbf{M} \in \mathbb{R}^{N \times N}$ used in Eq. (2) and Eq. (3) in the main paper can be computed as :

$$\mathbf{M}(i, j) = \mathbf{1}(\|\mathbf{p}_i - \mathbb{T}\mathbf{p}'_j\|_2 < \tau), \quad (1)$$

where $\mathbf{1}$ is the indicator function, \mathbf{p}_i and \mathbf{p}'_j are the 3D centers of the i th and j th local patches from the anchor point cloud \mathbf{P} and the positive point cloud \mathbf{P}' , respectively. \mathbb{T} is the applied synthetic transformation. τ is set to 0.5 in our experiments. Different from learning the local descriptors, the training of the global descriptor assembler can make use of the GPS/INS readings provided by the dataset as the supervision information. To this end, each downsampled submap is tagged with a UTM coordinate at its respective centroid. Similar point clouds are defined to be at most 10m apart and those dissimilar to be at least 50m apart.

4.3 Two-Phase Training

In practice, we train our model in two phases to improve stability. We first focus on the local descriptor matching task. The input is a pair of point clouds and the output is two sets of matching descriptors associated with their matching confidence. The loss function used is formulated as:

$$L = L_{desc} + \lambda L_{det}. \quad (2)$$

We use a batch size of 6 pairs of point clouds and for each pair of training samples, we randomly choose 512 points from the anchor point cloud as centers of the local patches (the points are chosen correspondingly in the transformed point clouds). This results in 6×512^2 combinations for the network per batch. We use the Adam optimizer with a learning rate of 10^{-4} , $\beta_1 = 0.9$, $\beta_2 = 0.999$. The learning rate is successively halved every 5 epochs. In the second phase, we freeze the local feature encoder so that the networks are enforced to train later feature extracting layers which aims to model higher-level semantic information, as well as the attention map and the NetVLAD layer which is for aggregating local features into discriminative global features. We use the same lazy quadruplet loss as used in PointNetVlad [1] and PCAN [10] which is formulated as:

$$\begin{aligned} L_{lazyQuad}(\mathcal{T}) = & \max([\alpha + \delta_{pos} - \delta_{neg}]_+) \\ & + \max([\beta + \delta_{pos} - \delta_{neg*}]_+), \end{aligned} \quad (3)$$

where \mathcal{T} refer to a tuple of point cloud samples in one training iteration, which includes an anchor point cloud \mathbf{P}_{anc} , a set of positive and a negative point clouds to the anchor $\{\mathbf{P}_{pos}\}, \{\mathbf{P}_{neg}\}$ as well as a random sample \mathbf{P}_{neg*} that is dissimilar to all the former samples. We use $\delta_{pos/neg}$ to denote the L2-distance between the global descriptor vectors of \mathbf{P}_{anc} and $\mathbf{P}_{pos/neg}$, while δ_{neg*} measures the distance between \mathbf{P}_{neg*} and \mathbf{P}_{neg} . The max operator of the first term selects the best positive in $\{\mathbf{P}_{pos}\}$, and the hardest negative in $\{\mathbf{P}_{neg}\}$. Similarly, the hardest negative in $\{\mathbf{P}_{neg}\}$ that gives the smallest δ_{neg*} value is selected by the second loss term. α and β are two different constant parameters giving the margins which are set as 0.5 and 0.2 respectively in our experiments. In this stage, we train our model with a single batch of data consisting 1 \mathbf{P}_{anc} , 1 \mathbf{P}_{neg*} , 2 \mathbf{P}_{pos} and 8 \mathbf{P}_{neg} . The Adam optimizer is used with $\beta_1 = 0.9$, $\beta_2 = 0.999$. The

initial learning rate is set as 5×10^{-4} and exponentially decayed after every 10 epochs until 10^{-5} . The dimension of the output global descriptor and the number of clusters in the NetVLAD layer are set to be same as in [1, 10], i.e., 256 and 64, respectively.

4.4 Testing Set Generation by Stereo DSO

In Sec. 4.4 of the main paper, we use the point clouds generated by Stereo DSO [8] to evaluate the generalization capability of our method. Here we provide more details on the testing set used in our experiments. The Bumblebee XB3 images under the wide-baseline configuration from the Oxford RobotCar dataset are taken as input to Stereo DSO, which estimates the camera poses and pixel depths by minimizing a photometric error obtained by direct image alignment. To extract a local point cloud, Stereo DSO is first run on a sub-sequence of images and all the valid 3D points associated with each keyframe are collected. These points are then projected into one common coordinate system utilizing the estimated relative camera poses.

To cover a wide range of day-time and weather conditions, eight sequences are chosen as listed in Tab. 1. One sequence in the overcast condition is set as the reference traversal, where the Stereo DSO is run on the full sequence. For each of the other seven sequences, 50 different images are uniformly selected as the starting frames and the corresponding local point clouds are extracted using the method described above. After filtering out invalid ones due to failures of Stereo DSO under some challenging situations, we obtain 332 point clouds as shown in Tab. 1. To generate the ground truth transformations, the corresponding point clouds accumulated from LiDAR scans are also collected to calculate the accurate relative poses by performing ICP between overlapped point clouds. This gives us 318 pairwise poses for testing. We crop each extracted point cloud with a fixed radius of 30m around its centroid followed by a downsampling step with a VoxelGrid filter with the fixed grid size of 0.2m. Lastly, we randomly rotate each point cloud around the vertical axis to evaluate the rotational invariance. We thank the authors of Stereo DSO again for their great help on generating the point clouds for us. These point clouds as well as the related data splits explained in Sec. 4.2 will be released to the community upon the publication of this paper.

4.5 Weak Supervision Used in the Ablation Study

In the ablation study, we compare our local descriptors with those trained using a weak supervised manner proposed by 3DFeatNet [3]. The weak supervision information is provided at the submap-level which is based on the ground truth coarse registrations of the point clouds in the training set. More specifically, the input to the network in each iteration is an anchor point cloud \mathbf{P}_{anc} with a positive point cloud \mathbf{P}_{pos} and a negative point cloud \mathbf{P}_{neg} to the anchor. The model is trained using the triplet loss computed by :

Sequence	Conditions	Valid point clouds
2014-12-02-15-30-08 (ref)	overcast	107
2014-11-14-16-34-33	night	8
2014-12-16-18-44-24	night	17
2014-11-25-09-18-32	rain	22
2015-02-03-08-45-10	snow	39
2015-03-24-13-47-33	sunny	43
2015-02-20-16-34-06	dusk, roadworks	47
2015-11-10-14-15-57	overcast, roadworks	49

Table 1: Number of point clouds generated by Stereo DSO for point cloud registration evaluation.

$$L = \sum_{n=1}^N \left[\min_{\mathbf{x}_i \in \mathcal{X}_{pos}} \|\mathbf{x}_n - \mathbf{x}_i\|_2 - \min_{\mathbf{x}_j \in \mathcal{X}_{neg}} \|\mathbf{x}_n - \mathbf{x}_j\|_2 + \gamma \right]_+, \quad (4)$$

where N is the number of local clusters extracted in each point cloud. Each local descriptor \mathbf{x}_n in \mathcal{P}_{anc} can find its closest descriptor in both \mathcal{P}_{pos} and \mathcal{P}_{neg} . The former is considered as the correct local match and the latter one is considered as the hardest negative. γ is the margin which is set as 0.2 in our experiments.

References

1. Angelina Uy, M., Hee Lee, G.: PointNetVLAD: Deep point cloud based retrieval for large-scale place recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4470–4479 (2018)
2. Groh, F., Wieschollek, P., Lensch, H.P.: Flex-convolution. In: Asian Conference on Computer Vision. pp. 105–122. Springer (2018)
3. Jian Yew, Z., Hee Lee, G.: 3DFeat-Net: Weakly supervised local 3D features for point cloud registration. In: Proceedings of the European Conference on Computer Vision. pp. 607–623 (2018)
4. Li, J., Lee, G.H.: USIP: Unsupervised stable interest point detection from 3D point clouds. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 361–370 (2019)
5. Li, Y., Bu, R., Sun, M., Wu, W., Di, X., Chen, B.: PointCNN: Convolution on x -transformed points. In: Advances in Neural Information Processing Systems. pp. 820–830 (2018)
6. Maddern, W., Pascoe, G., Linegar, C., Newman, P.: 1 year, 1000 km: The Oxford RobotCar dataset. The International Journal of Robotics Research **36**(1), 3–15 (2017)
7. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: PointNet++: Deep hierarchical feature learning on point sets in a metric space. In: Advances in Neural Information Processing Systems. pp. 5099–5108 (2017)
8. Wang, R., Schworer, M., Cremers, D.: Stereo DSO: Large-scale direct sparse visual odometry with stereo cameras. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 3903–3911 (2017)
9. Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M.: Dynamic graph CNN for learning on point clouds. ACM Transactions on Graphics (TOG) **38**(5), 1–12 (2019)
10. Zhang, W., Xiao, C.: PCAN: 3D attention map learning using contextual information for point cloud based retrieval. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 12436–12445 (2019)